UNCLASSIFIED

AD NUMBER
ADB023903
LIMITATION CHANGES
TO: **Approved for public release; distribution is unlimited.**
FROM: **Distribution authorized to U.S. Gov't. agencies only; Test and Evaluation; JUN 1977. Other requests shall be referred to Air Force Armament Lab., Eglin AFB, FL.**
AUTHORITY
USADTC ltr 1 Oct 1980

THIS PAGE IS UNCLASSIFIED

THIS REPORT HAS BEEN DELIMITED AND CLEARED FOR PUBLIC RELEASE UNDER DOD DIRECTIVE 5200.20 AND NO RESTRICTIONS ARE IMPOSED UPON ITS USE AND DISCLOSURE.

DISTRIBUTION STATEMENT A

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AFATL-TR-77-85

AN AERIAL GUNNERY AERODYNAMIC LEAD PURSUIT MODEL

THE DIVISION OF ENGINEERING RESEARCH
LOUISIANA STATE UNIVERSITY
BATON ROUGE, LOUISIANA 70803

JUNE 1977

FINAL REPORT FOR PERIOD
JUNE 1976-JUNE 1977

Distribution limited to U. S. Government agencies only; this report documents test and evaluation; distribution limitation applied June 1977. Other requests for this document must be referred to the Air Force Armament Laboratory (DLYD), Eglin Air Force Base, Florida 32542.

AIR FORCE ARMAMENT LABORATORY

AIR FORCE SYSTEMS COMMAND • UNITED STATES AIR FORCE

EGLIN AIR FORCE BASE, FLORIDA

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER AFATL-TR-77-85	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) AN AERIAL GUNNERY AERODYNAMIC LEAD PURSUIT MODEL		5. TYPE OF REPORT & PERIOD COVERED Final Report. June 1976 — June 1977
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) A. J. McPhate D. R. Fleming		8. CONTRACT OR GRANT NUMBER(s) F08635-76-C-0273
9. PERFORMING ORGANIZATION NAME AND ADDRESS The Division of Engineering Research, Louisiana State University Baton Rouge, Louisiana 70803		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS JON: 9134-07-03 Program Element 62602F
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Armament Laboratory Armament Development and Test Center Eglin Air Force Base, Florida 32542		12. REPORT DATE June 1977
		13. NUMBER OF PAGES 90
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution limited to U. S. Government agencies only; this report documents test and evaluation; distribution limitation applied June 1977. Other requests for this document must be referred to the Air Force Armament Laboratory (DLYD), Eglin Air Force Base, Florida 32542.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Available in DDC

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Aerial Gunnery Aerodynamic Lead Pursuit Model

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A digital simulation of aerial gunnery was developed. A six-degree-of-freedom attacking aircraft controlled by control surface deflections was implemented to perform aerodynamic lead pursuit of an evading target. The attacker is capable of flying within the bounds of its performance envelope. The target is a coordinated flight and aerodynamically constrained model capable of some evasive logic. Three sight systems have been implemented into the simulation. The program was written in FORTRAN IV and has been checked out on the XDS Sigma 5, the IBM 360/65, and the CDC 6600 computers.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

389 061

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

/

UNCLASSIFIED

PREFACE

This report was prepared by the Division of Engineering Research, Louisiana State University, Baton Rouge, Louisiana 70803 under Contract F08635-76-C-0273, with the Air Force Armament Laboratory, Armament Development and Test Center, Eglin Air Force Base, Florida 32542. This effort began in June 1976 and was completed in June 1977. Mr. Gerald Solomon (DLYD) monitored the program for the Armament Laboratory.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER:

J R Murray

J. R. MURRAY
Chief, Analysis Division

(The reverse of this page is blank.)

TABLE OF CONTENTS

TABLE OF CONTENTS (CONCLUDED)

LIST OF FIGURES

(The reverse of this page is blank)

SECTION I

INTRODUCTION

OBJECTIVES

The design intent of the effort described by this report was the construction of a digital software simulation of an attacker-target encounter in aerial gunnery combat. The simulation was restricted in scope to the target tracking phase of the encounter with no particular attention being focused on tactical aspects. The primary effort was expended in obtaining an optimally controlled attacking aircraft capable of flying within the boundaries of its performance envelope, the emphasis being placed upon obtaining realistic aerodynamic lead pursuit in the tracking phase of the encounter. Complimentary to this effort were the simulation of the target and the sight systems and the overall problem of fitting the simulation into GAMES (Reference 1).

LITERATURE SURVEY

Consistent with the proposed work outline, a literature survey of available simulations was made to determine the basic capabilities of those simulations (References 2 and 3). Based upon the requirement that the attacker be able to fly realistic uncoordinated flight in the tracking phase, the IMAGE simulation was the only viable candidate for the attacker airframe simulation. The development of a totally new simulation was considered but was deemed infeasible within the time constraints allowed for the project. Thus, the decision to extract from the IMAGE simulation the airframe dynamics and aerodynamics was made and implemented.

OVERVIEW

The overall effort was divided into four basic tasks: (1) Attacker airframe and pilot, (2) target simulation, (3) sight systems, and (4) implementation of tasks 1, 2, and 3 into GAMES.

Attacker Simulation

Section II of this report is devoted to the description of the attacker simulation. The attacker is a full six-degree-of-freedom airframe model with a zero time lag pilot and control system. All data defining the attacker is read in from the input file and defines a three control surface airframe. The aerodynamics data is stored in table form, and a table lookup scheme is used to extract information.

Target Simulation

The target simulation is described in detail in Section III of this report. The target is a psuedo six-degree-of-freedom model in that coordinated flight is assumed and attitude is computed in a dependent fashion from the lineal velocity and acceleration. Acceleration is computed as a demand quantity or as an input quantity based upon the type of maneuver specified.

Target maneuvers are defined by preprogramming a battery of sequentially activated maneuvers. Up to 10 steps or maneuvers may be programmed to define the target motion profile with the sequencing being influenced by what the attacker and target do. Tabular data is also required to define the performance envelope of the target.

Sight Systems

The sight systems have been implemented in subroutine format, and first and second order perfect sights as well as mechanization of one other sight have been structured. Section IV of this report gives the details of the sight system implementation.

GAMES

The final product of the effort is documented in Section V of this report. The attacker-pilot, target, and sight systems are imbedded into the GAMES program as a subunit that updates the state vector for both attacker and target on demand from the executive part of the simulation. GAMES itself has been minimally modified to accept this alternative to its present approach to defining the engagement.

SECTION II

ATTACKER

PURPOSE

The attacker simulation is a six-degree-of-freedom digital model of an airframe and a controller. The controller simulates a pilot whose aim is to maneuver the attacking airframe into firing position. Firing position is achieved by nulling the tracking errors between the line of sight to the target and the pipper position in the heads up display (HUD) computed by the sight system.

SUBPROGRAM PILOT

Subprogram PILOT is an optimum controller for the airframe simulated in the subroutine TURKEY. The task of PILOT is to maneuver, by setting the control surfaces, the airframe into the best position for a successful aerial engagement with the target. All control decisions are made utilizing current information about the attacker and the target, i.e., PILOT does not have access to information concerning the future conditions of either the airframe or the target.

PILOT utilizes the fact that the most rapid means of minimizing the integral of the error is to use maximal rectilinear and angular accelerations and decelerations for the airframe. PILOT communicates to TURKEY by means of a control vector composed of aileron setting, stabilator setting, thrust or throttle control setting, and rudder setting. Therefore, the individual components will, in general, be set at their limits while maneuvering to achieve near aerodynamic lead pursuit. Only in the final stages of the engagement, when the magnitude of the error and its first derivative is small, will the components of the control vector be at some intermediate value.

ERROR COMPUTATION

The error vector is composed of roll error, pitch error, range error, and yaw error. These quantities are computed from the elevation and traverse tracking errors and the range to the target (see Figure 1).

SINGLE CHANNEL CONTROLLER

PILOT uses four single channel controllers, each responsive to a single component of the error vector. Roll error controls the aileron deflections; pitch error controls the stabilator deflections. The thrust is controlled by the range error. Rudder deflections are controlled by the yaw error. In effect, the control vector is the sum of the output of each individual

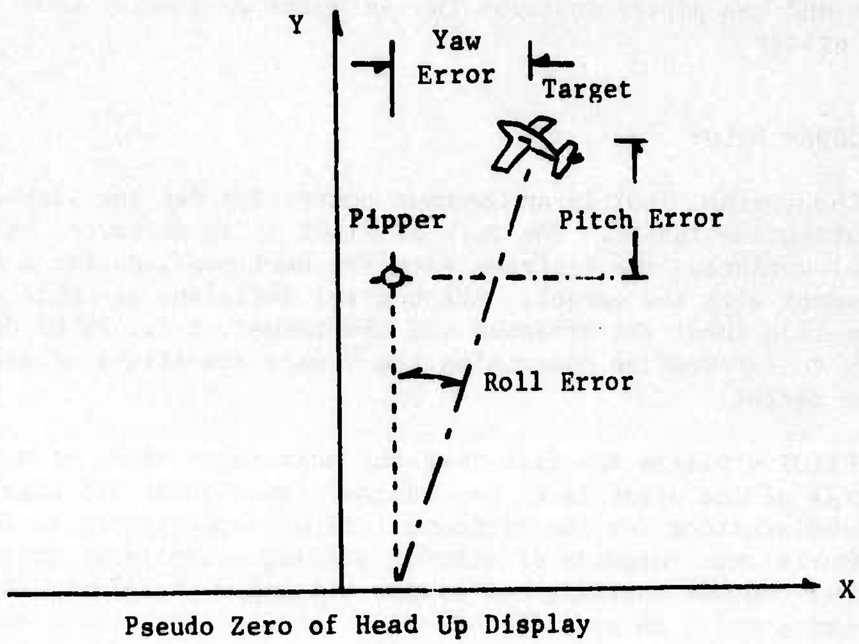

Figure 1. View Through Heads Up Display

controller. Each controller assumes that the other components of the control vector have a negligible effect on the error which it is trying to control.

CONTROLLER LOGIC

In PILOT each controller tries to minimize the integral of the error which it senses. This is accomplished by nulling the error and its first derivative as rapidly as possible. The controller treats the response of the airframe as an unknown transfer function. The controller, sensing the error and the first derivative of the error, is able to sample the range of second derivatives of the error by changing its respective component of the control vector. This enables the controller to choose an appropriate value for the control vector component.

The controller is at liberty to adjust its control parameter and is able to obtain a complete range of values for the second derivative of the error. In general, these will have extreme magnitudes when the control vector is at its limits.

<u>Control Decision</u> If the error at any time t be denoted by x, its first derivative is denoted by $\dot{x}$ and its second derivative is denoted by $\ddot{x}$. The controller has at its command all available values of $\ddot{x}$. The optimal choice will be, in general, between the maximal positive and the maximal negative values of $\ddot{x}$. These values are assumed to be constant over an interval in time. x_1 and $\dot{x}_1$ is the error and its first derivative at the start of the time interval. $\ddot{x}_1$ is judiciously chosen as the second derivative which will improve the situation over the next time interval, t_1. x_2 and $\dot{x}_2$ are the error and its first derivative at the end of the time interval t_1 and at the start of the second time interval t_2. $\ddot{x}_2$ is the second derivative of the error to be used during this second time interval, t_2. The magnitudes of t_1 and t_2 can be obtained from Equations (1) through (4)

$$1/2\ \ddot{x}_1 t_1^2 + \dot{x}_1 t_1 + x_1 = x_2 \tag{1}$$

$$\ddot{x}_1 t_1 + \dot{x}_1 = \dot{x}_2 \tag{2}$$

$$1/2\ \ddot{x}_2 t_2^2 + \dot{x}_2 t_2 + x_2 = 0 \tag{3}$$

$$\ddot{x}_2 t_2 + \dot{x}_2 = 0 \tag{4}$$

At the end of the second time interval, both the error and its first derivative will be zero if $\ddot{x}_1$ and $\ddot{x}_2$ were constant over the two time intervals.

This method will give reasonable results, but it is more economical,

since the simulation marches in time, to merely ask if the simulation has reached the end of the first interval. This is accomplished by assuming that the first interval is over and by using Equations (3) and (4) to evaluate the projected error at the end of the time interval t_2. t_2 is obtained from Equation (4).

$$t_2 = -\dot{x}_2/\ddot{x}_2 \qquad (5)$$

Equation (5) is obtained from Equation (4) by the requirement that at the end of the interval t_2 the derivative of the error should be zero. The projected error is then calculated.

If the projected error is sufficiently small, then the assumption that the end of t_1 has been reached is correct. The component of the control vector is set such that the second derivative is $\ddot{x}_2$.

If the magnitude of the projected error is greater than specified limits and if the value is of the same sign as the present value of the error, the end of the interval t_1 has not been reached. In this case the control vector should be set such that the second derivative is equal to $\ddot{x}_1$.

However, if the projected error at t_2 has the opposite sign of its present value, then the airframe is predicted to overshoot the desired state. This means that the roles of $\ddot{x}_1$ and $\ddot{x}_2$ must be interchanged. The component of the control vector again should cause the second derivative of the state vector to equal the new $\ddot{x}_1$.

The algorithm used in choosing the proper extreme second derivatives is shown in the flow chart in the Figure 2. The assumption that $\ddot{x}_1$ and $\ddot{x}_2$ are constant is permissible because they are updated, and the above test is made frequently.

TOTAL CONTROLLER

The assumption that other components of the control vector do not noticeably affect the error sensed by any individual controller is permissible because the errors and their respective derivatives are updated, and adjustments to the control vector are made frequently.

If the magnitude of the component of the error vector and the magnitude of the derivative of that error which a controller is sensing are sufficiently small, that component of the control vector is attenuated.

If the total error, pitch error plus the yaw error, is large, the rudder is not operational. However, if the error is sufficiently small, the rudder is used and the aileron deflections are set to zero.

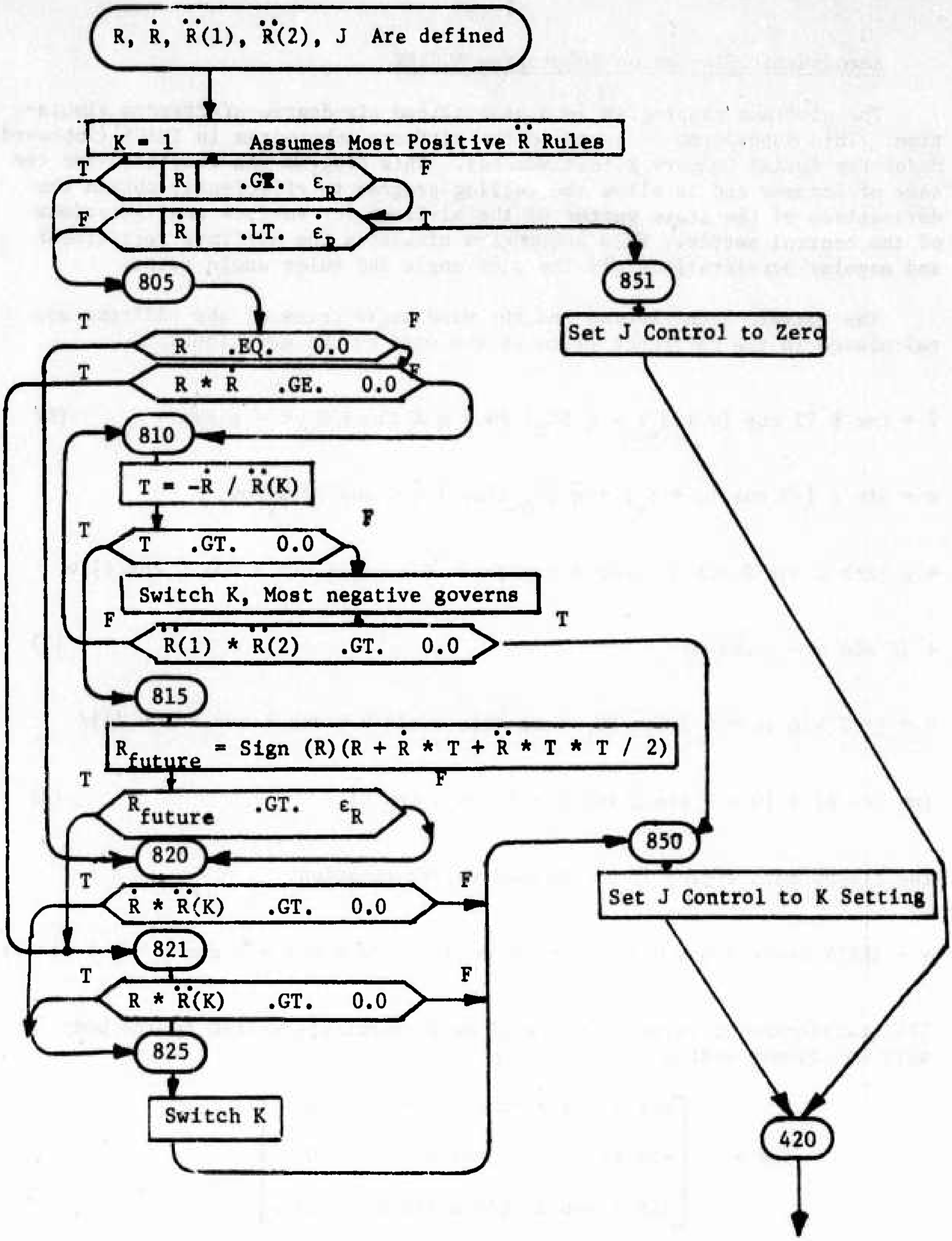

Figure 2. Flow Chart of Controller Logic

Aerodynamic Simulation Subprogram TURKEY

The airframe subprogram is a generalized six-degree-of-freedom simulation. This subprogram is based on the airframe subprogram in IMAGE (Improved Model for Aerial Gunnery Effectiveness). This program was rewritten for the sake of economy and to allow the calling program to efficiently obtain the derivatives of the state vector of the airframe for various configurations of the control vector. This subprogram simulates the airframe rectilinear and angular accelerations and the wind angle and euler angle rates.

The forward acceleration and the wind angle rates of the airframe are calculated in the reference frame of the wind by the equations:

$$\dot{v} = \cos\beta\,[T\cos(\alpha+\delta_e) - \bar{q}\,SC_D]/m + \bar{q}\,S\sin\beta\,C_y/m - g\sin\gamma \tag{6}$$

$$\dot{\beta} = \sin\beta\,[-T\cos(\alpha+\delta_e) + \bar{q}\,SC_D\,1/mv + \bar{q}\,S\cos\beta]\,C_y/mv$$

$$+ g\,[\cos\alpha\sin\beta\sin\theta + \cos\beta\cos\theta\sin\phi - \sin\alpha\sin\beta\cos\theta\cos\phi]/v$$

$$+ (P\sin\alpha - R\cos\alpha) \tag{7}$$

$$\dot{\alpha} = [-T\sin(\alpha+\delta_e) - \bar{q}\,SC_L + mg\,(\sin\alpha\sin\theta + \cos\alpha\cos\theta\cos\phi)]/$$

$$(mv\cos\beta) + [Q - R\sin\alpha\tan\beta - P\cos\alpha\tan\beta] \tag{8}$$

The flight path angle, γ, is defined by the equation:

$$\gamma = \mathrm{ARSIN}\,(-\sin\phi\cos\theta\sin\beta - \cos\beta\sin\alpha\cos\theta\cos\phi + \cos\alpha\cos\beta\sin\theta) \tag{9}$$

The transformation relating the wind axis coordinate system to the body axis coordinate system is defined as:

$$TWB = \begin{bmatrix} \cos\alpha\cos\beta & -\cos\alpha\sin\beta & \sin\alpha \\ \sin\beta & \cos\beta & 0 \\ \sin\alpha\cos\beta & -\sin\alpha\sin\beta & -\cos\alpha \end{bmatrix}$$

The angular accelerations and euler angle rates are calculated in the body axis coordinated system by the equations:

$$\dot{P} = \frac{I_{yy} - I_{zz}}{I_{xx}} QR + \frac{I_{xz}}{I_{xx}} (\dot{R} + PQ) + \frac{\bar{q}\,SB}{I_{xx}} [\cos\beta \cos\alpha\, C_\ell - \sin\alpha\, C_n] \tag{10}$$

$$\dot{Q} = \frac{I_{zz} - I_{xx}}{I_{yy}} RP + \frac{I_{xz}}{I_{yy}} (R^2 - P^2) + \frac{\bar{q}Sc}{I_{yy}} C_m + M_t T/I_{yy} \tag{11}$$

$$\dot{R} = \frac{I_{xx} - I_{yy}}{I_{zz}} PQ + \frac{I_{xz}}{I_{zz}} [\dot{P} - QR] + \frac{\bar{q}Sb}{I_{zz}} [\cos\beta \sin\alpha\, C_\ell + \cos\alpha\, C_n] \tag{12}$$

$$\dot{\phi} = P + \dot{\psi} \sin\Theta \tag{13}$$

$$\dot{\Theta} = Q \cos\phi - R \sin\phi \tag{14}$$

$$\psi = (R \cos\phi + Q \sin\phi)/ \cos\Theta \tag{15}$$

The transformation relating the body axis coordinate system to the inertial coordinate system is defined as:

$$TBE = \begin{bmatrix} \cos\psi\cos\Theta & \sin\psi\cos\Theta & -\sin\Theta \\ \cos\psi\sin\Theta\sin\phi - \sin\psi\cos\phi & \sin\psi\sin\Theta\sin\phi + \cos\psi\cos\phi & \cos\Theta\sin\phi \\ \cos\psi\sin\Theta\cos\phi + \sin\psi\sin\phi & \sin\psi\sin\Theta\cos\psi - \cos\psi\sin\phi & \cos\Theta\cos\phi \end{bmatrix} \tag{16}$$

The rectilinear velocities and accelerations in the inertial reference frame are obtained by the transformation of the velocity vector and its derivative from the wind axis coordinate system into the inertial reference frame.

$$v_E = [T_{EB}][T_{WB}]v_w \qquad \text{where } v_w = \begin{bmatrix} v \\ o \\ o \end{bmatrix} \tag{17}$$

$$\dot{v}_E = A_E = [\dot{T}_{EB}][T_{WB}]v_w + [T_{EB}][\dot{T}_{WB}]v_w + [T_{EB}][T_{WB}]\dot{v}_w \tag{18}$$

where

$$\frac{d}{dt}[T_{WB}] = \begin{bmatrix} -\sin\alpha\cos\beta\,\dot{\alpha} & -\sin\alpha\sin\beta\,\dot{\alpha} & \cos\alpha\,\dot{\alpha} \\ -\cos\alpha\sin\beta\dot{\beta} & +\cos\alpha\cos\beta\dot{\beta} & \\ \cos\beta\dot{\beta} & -\sin\beta\dot{\beta} & 0 \\ \cos\alpha\cos\beta\dot{\alpha} & -\cos\alpha\sin\beta\dot{\alpha} & \\ -\sin\alpha\sin\beta\dot{\beta} & -\sin\alpha\cos\beta\dot{\beta} & +\sin\alpha\dot{\alpha} \end{bmatrix}$$

and

$$\frac{d}{dt}[T_{BE}] = [T_{BE}] \begin{bmatrix} 0 & -R & +Q \\ +R & 0 & -P \\ -Q & +P & 0 \end{bmatrix}$$

The airframe simulation subprogram is constructed so that the recalculation of the derivative of the state vector is done efficiently for various control vectors. The majority of the calculations not involving the control vector are saved in dummy variables. Figure 3 is a flow diagram of the airframe simulation.

Data for the Attacker The data to describe the attacker is read by subprograms PILOT and TURKEY. In the read sequence PILOT reads first, then calls ESCAPE which then reads its data to describe the target. ESCAPE is described in Section III of this report. Subsequent to ESCAPE reading data, the SSIGHT subprogram reads data as defined in Section IV of this report. Finally TURKEY reads data to define the airframe.

Data Read by PILOT PILOT reads a minimal amount of data -- only that necessary to define the integration, control, print out, and initial conditions of the attacker.

CARD	FORMAT	QUANTITIES READ
1	(A10, 2I5, E10.2)	LABEL, NF1, NF9, DTIME
2,3,4	(A10,7E10.2)/10X,7E10.2)	LABEL, (SV(I),I=1,15)

The quantities read from the four cards are:

LABEL	Any identifying label up to 10 characters long.
NF1	Control frequency indicator. Sets the number of integration steps between control cycles.

NF9	Print out indicator. Sets the number of control cycles between printout by PILOT.
DTIME	The integration step size to be used by the Runge-Kutta integration algorithm.
SV(1)	X_1 position of attacker
SV(2)	X_2 position of attacker
SV(3)	X_3 position of attacker
SV(4)	$\dot{X}_1$
SV(5)	$\dot{X}_2$
SV(6)	$\dot{X}_3$
SV(7)	$(\dot{X}_1^2 + \dot{X}_2^2 + \dot{X}_3^2)^{1/2}$
SV(8)	Roll Euler Angle, ϕ
SV(9)	Pitch Euler Angle, θ
SV(10)	Heading Euler Angle, Ψ
SV(11)	Angle of Attack, α
SV(12)	Side Slip Angle, β
SV(13)	Angular Velocity, Roll Axis
SV(14)	Angular Velocity, Pitch Axis
SV(15)	Angular Velocity, Yaw Axis

<u>Data Read by TURKEY</u> All of the aerodynamic performance tables used by TURKEY to simulate the airframe are read on the first call to TURKEY. Four separate reads are involved for the four COMMON arrays: IDUM1, TDUM1, TDUM2, and DUM. IDUM1, TDUM1, TDUM2, and DUM are only read once. The read sequence is IDUM1, TDUM1, TDUM2, and then DUM. IDUM1 is integer, and contains 131 integers that describe the data made up of TDUM1 and TDUM2. TDUM1 and TDUM2 store REAL data and have 550 and 4700 words, respectively. IDUM1, TDUM1, and TDUM2 are punched into cards by the data input program from the modified IMAGE program described by Reference 5.

Array DUM contains the following data:

DUM	DATA
1	Angular orientation of the thrust vector
2	Wing surface area
3	Wing span
4	Mass of airframe, slugs
5	Acceleration of gravity
6	Mean aerodynamic chord length
7	Moment of inertia XX, slug ft^2
8	Moment of inertia YY, slug ft^2
9	Moment of inertia ZZ, slug ft^2
10	Product of inertia XZ, slug ft^2
11	Angle of wing with respect to body
12	Moment arm of thrust vector
13	Rudder upper limit, degrees
14	Rudder lower limit, degrees
15	Stabilator upper limit, degrees
16	Stabilator lower limit, degrees
17	Aeleron upper limit, degrees
18	Aeleron lower limit, degrees
19	Thrust value 1, pounds
20	Thrust value 2, pounds
21	Thrust value 3, pounds
22	Thrust value 4, pounds
23	Maximum angle of attack, degrees
24	Maximum side slip, degrees
25	Maximum normal acceleration, G's.

DUM is read under a FORMAT of (8E10.2). Subsequent to the initial read of DUM, selected parts of DUM are re-read for each initialization of a new case, including the first. The re-read uses the same format, and the quantities re-read are DUM(4), DUM(7), DUM(8), DUM(9), DUM(10), which are the inertia parameters.

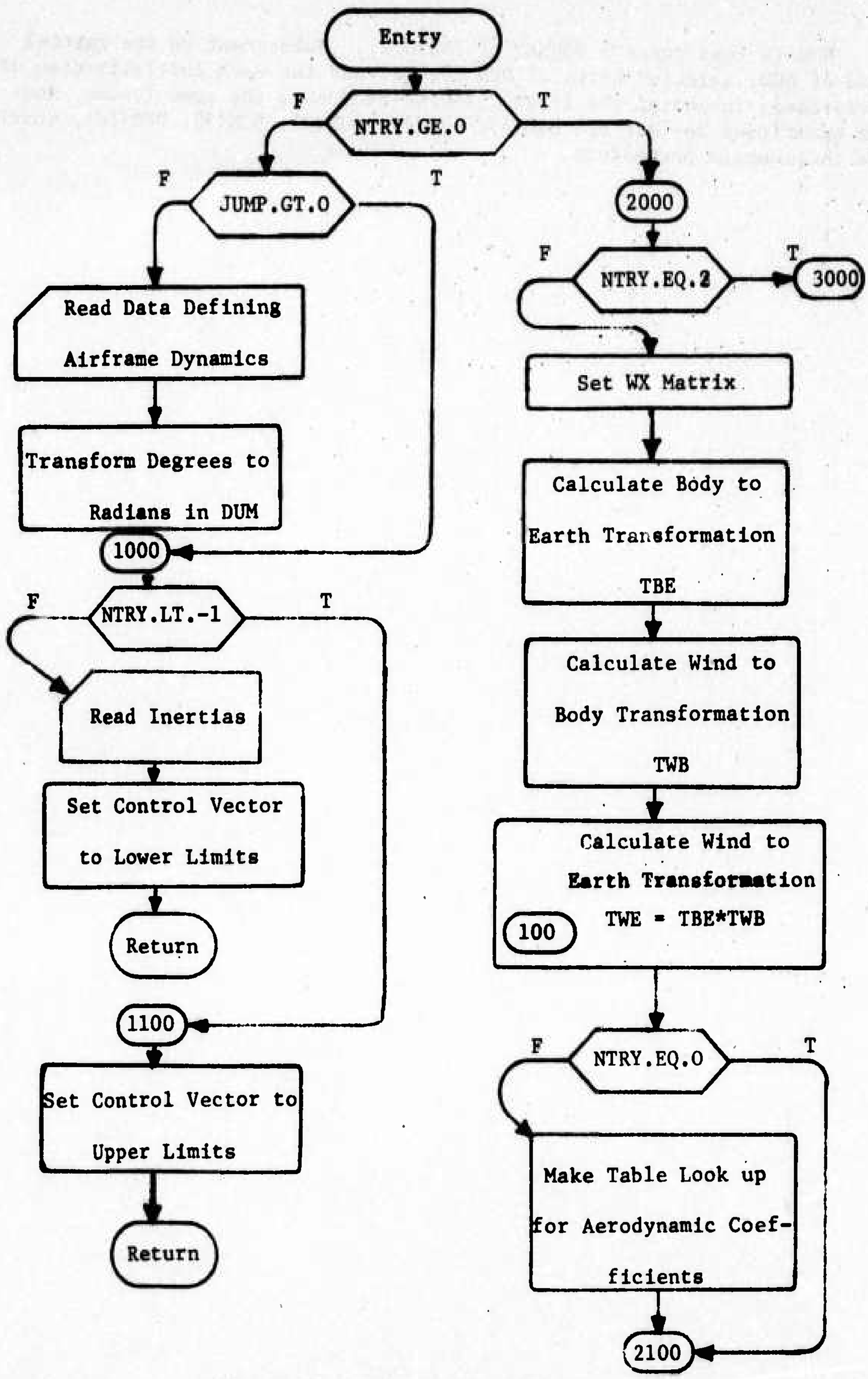

Figure 3. Airframe Simulation

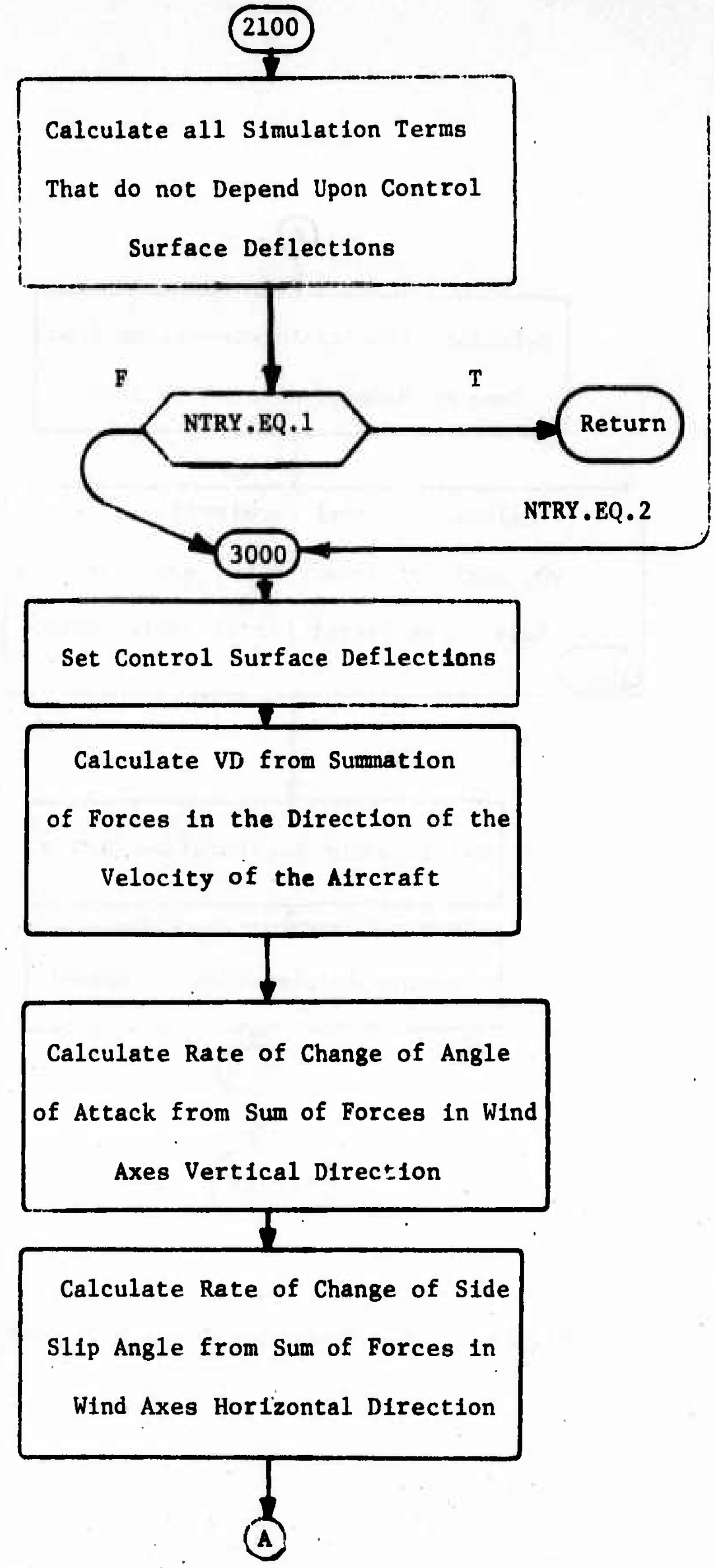

Figure 3. Airframe Simulation (Continued)

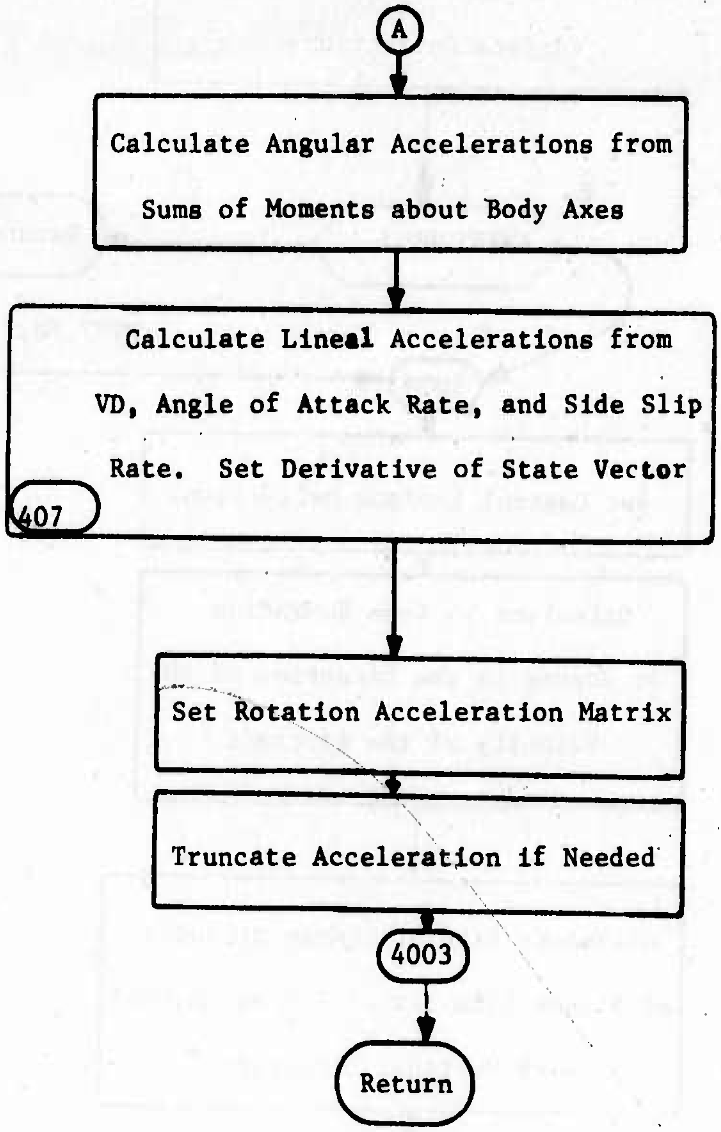

Figure 3. Airframe Simulation (Concluded)

SECTION III

TARGET EVASIVE MANEUVERS

INTRODUCTION

Subroutine ESCAPE was written to simulate the motion and decision of the target aircraft. The target defined by ESCAPE is a six-degree-of-freedom aircraft with realistic aerodynamic characteristics. Basically, the target flies coordinated flight along a flight trajectory that has been predetermined in a generalized fashion by the input data.

Within the target simulation, note is made of the state of both target and attacker, and the target makes evasive decisions based upon those states and bounds established by the input data. A serial decision sequence with no branching is defined such that at each decision point the simulation decides to either continue with the presently defined maneuver or to proceed to the next maneuver in the program stack of maneuvers. If the target completely exhausts the program stack of maneuvers, it simply vanishes by exiting to infinity.

Within the above concept, the analyst is able to program the target to the extent of defining the sequence of maneuvers in the program stack and defining the attacker/target state vector limits that dictate the termination of each step of the target program. There are seven basic maneuvers defined for the analyst to use to program the target. These are described below. Any component of the target or attacker state vector may be used as an upper or lower operating limit for each step of the program. A detailed listing of the index numbers of the state vector components is also given below.

TARGET MANEUVERS

Target maneuvers are designated by their index number, IZG, with integer value 1 through 7. The following gives a detail description of each maneuver:

IZG	Maneuver
1	Fly toward a specified point in space making changes in direction at a specified acceleration. This maneuver can be used to cause the target to fly a straight and level course as well as a circle in some circumstances.

IZG	Maneuver
2	Fly toward a point specified in space as an increment away from the target's position at the beginning of this maneuver. This maneuver becomes maneuver 1 after the new point is defined.
3	Fly toward a moving point a specified increment away from the target using a specified acceleration. This maneuver differs from 1 only in that the point always moves with the attacker. In the limit, this maneuver will result in a target velocity vector in the direction of the incremental position vector.
4	Change the direction of the target velocity vector to a new specified direction in a specified amount of time.
5	Make an incremental change in velocity vector direction where the increment is specified and the time to change is specified.
6	Chase an incremental change in velocity direction. The change is a constant, and the limiting velocity is parallel to the change vector.
7	Fly with a specified roll angle and normal acceleration. A linear function of time may be specified as the roll program for this maneuver.

EQUATIONS OF MOTION

For IZG = 1, 2, 3 the target is attempting a change in position. Over an interval of time, Δt, a change in position, ΔX, would be accomplished with constant kinematic acceleration according to

$$\Delta X = 1/2\ A_K \Delta t^2 \text{ or } A_K = \frac{2\Delta X}{\Delta t^2} \tag{19}$$

where ΔX and A_K are both vectors in the earth system.

The sensible acceleration would then be given by

$$A_S = A_K + G \tag{20}$$

where A_S and G are vectors, and specifically G is the vector representation of the acceleration of gravity.

The target is assumed to have small angle of attack so that the normal acceleration is considered normal to the wind or the target velocity vector. As such, the normal direction, N, is defined by the vector product $(V \times A_S) \times V$ where V is the velocity vector for the target. To achieve an acceleration of A_S then requires a normal acceleration, A_N, given by

$$A_N = A_S \cdot N \tag{21}$$

In the event that the calculated normal acceleration is in excess of the maximum allowed, the normal component of acceleration is truncated and the normal direction maintained.

The tangential component of acceleration is calculated by

$$A_T = A_S \cdot V/|V| \tag{22}$$

where V is the vector velocity of the target and $|V|$ is the scalar speed of the target. If this acceleration is not within the performance limits, it is truncated and the simulation continued.

For IZG = 4, 5, 6, the acceleration is computed for a change in target velocity rather than a change in target position. To this end

$$A_K = \frac{\Delta V}{\Delta t} \tag{23}$$

where ΔV is the vector change in velocity to occur over Δt change in time. The same analysis and constraints for A_N and A_T are used in cases 4, 5, and 6 as were used in cases 1, 2, and 3.

IZG = 7 is essentially a maneuver where the roll angle, and therefore N, is specified as a control. A_N is specified rather than being calculated. A_T is specified by means of throttle setting. Using A_T, A_N, and the roll angle, the kinematic acceleration is established and the motion of the target set.

INTEGRATION OF EQUATIONS OF MOTION

In all maneuver cases a program of acceleration is established for a small interval of time. The equation for velocity then becomes

$$V(t) = V(t_o) + A_K(t_o)(t-t_o) \tag{24}$$

and for the displacement

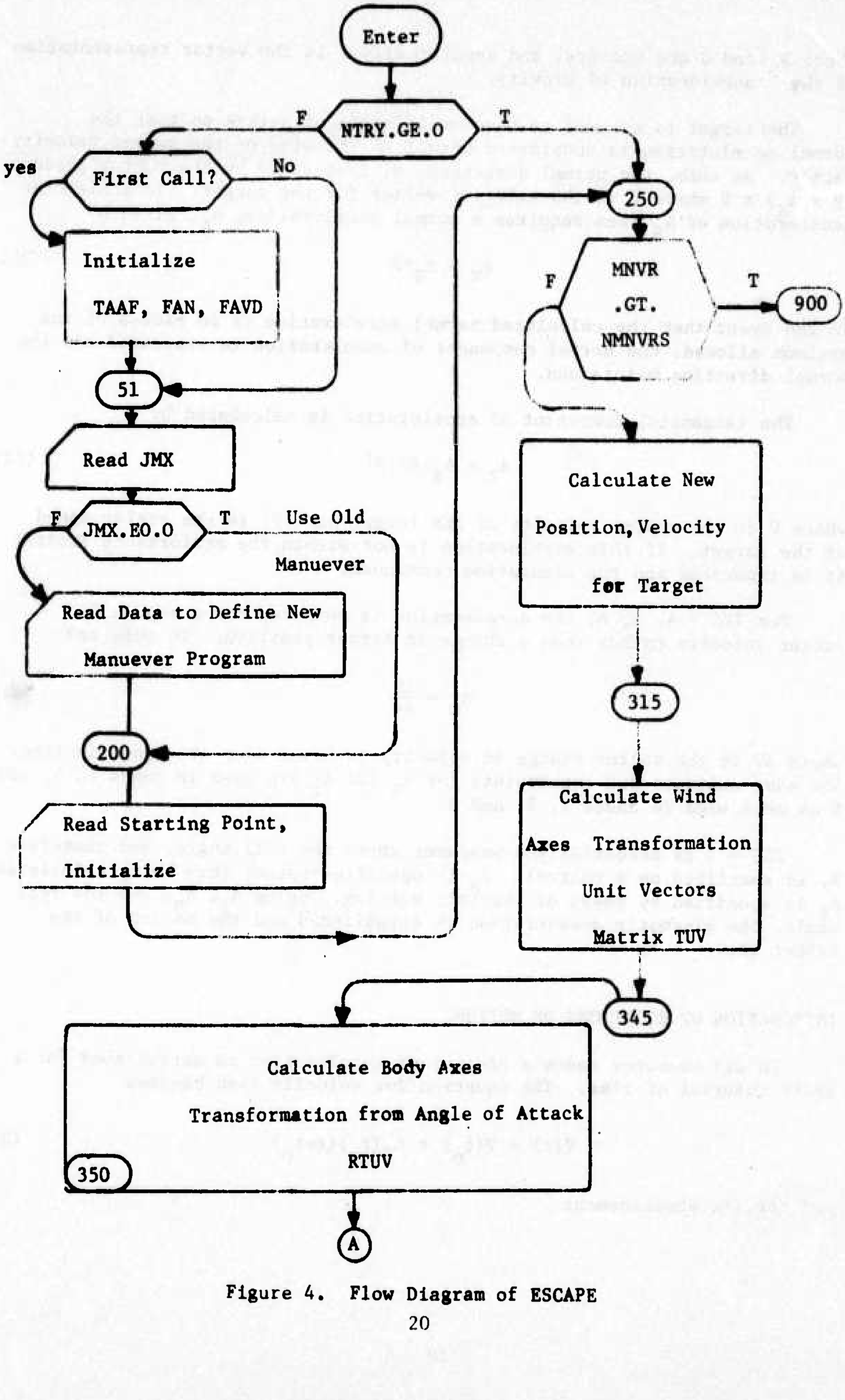

Figure 4. Flow Diagram of ESCAPE

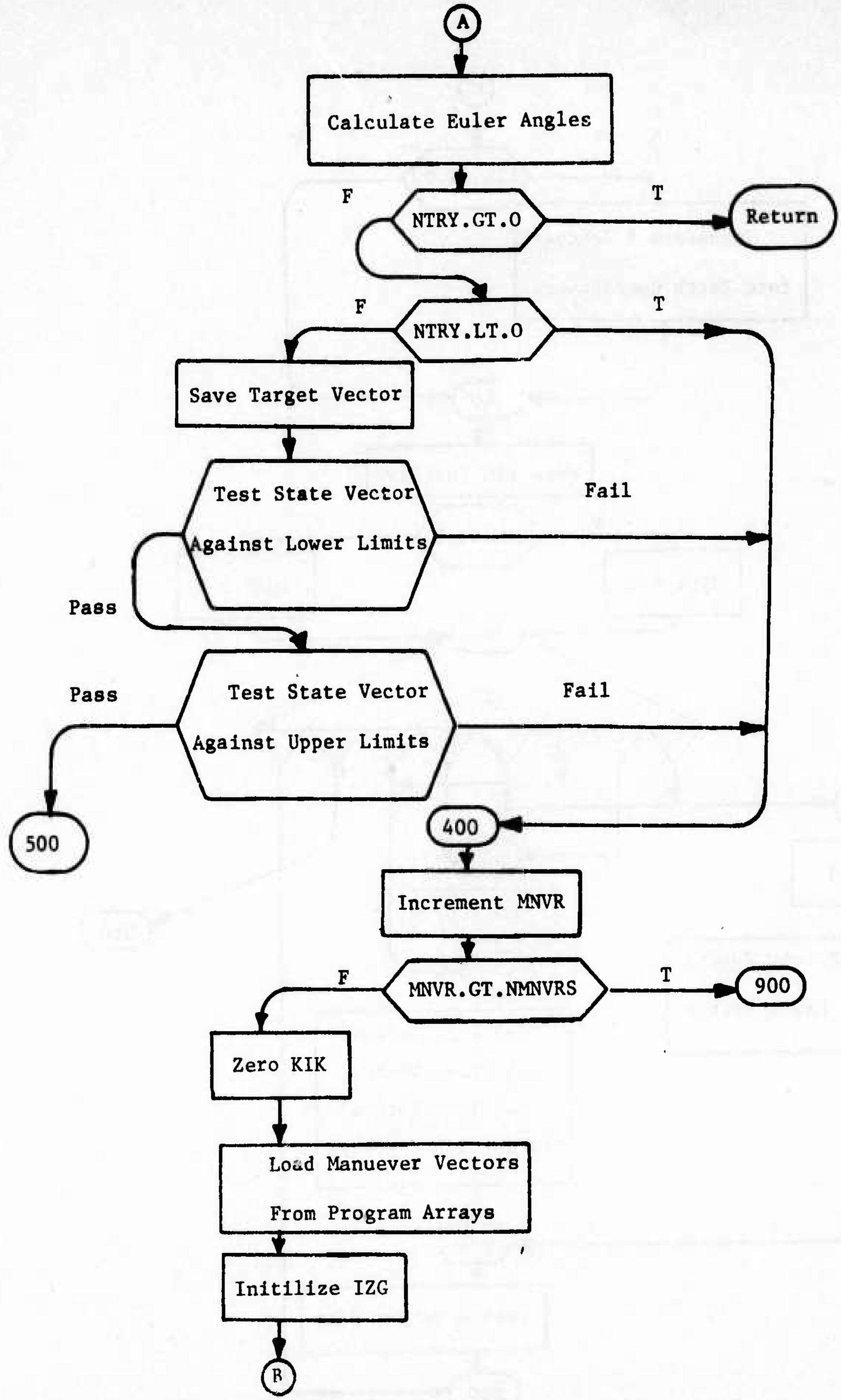

Figure 4. Flow Diagram of ESCAPE (Continued)

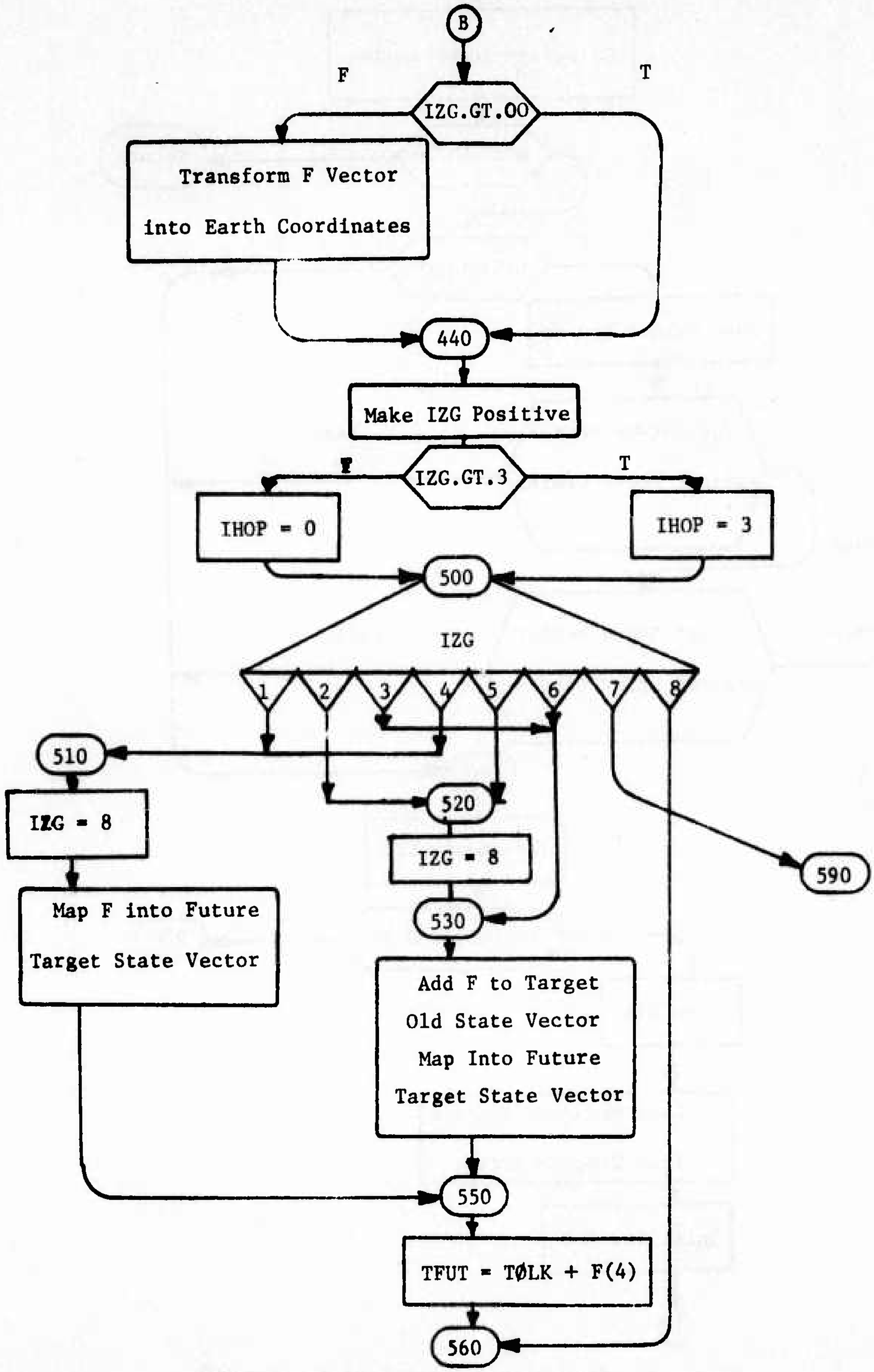

Figure 4. Flow Diagram of ESCAPE (Continued)

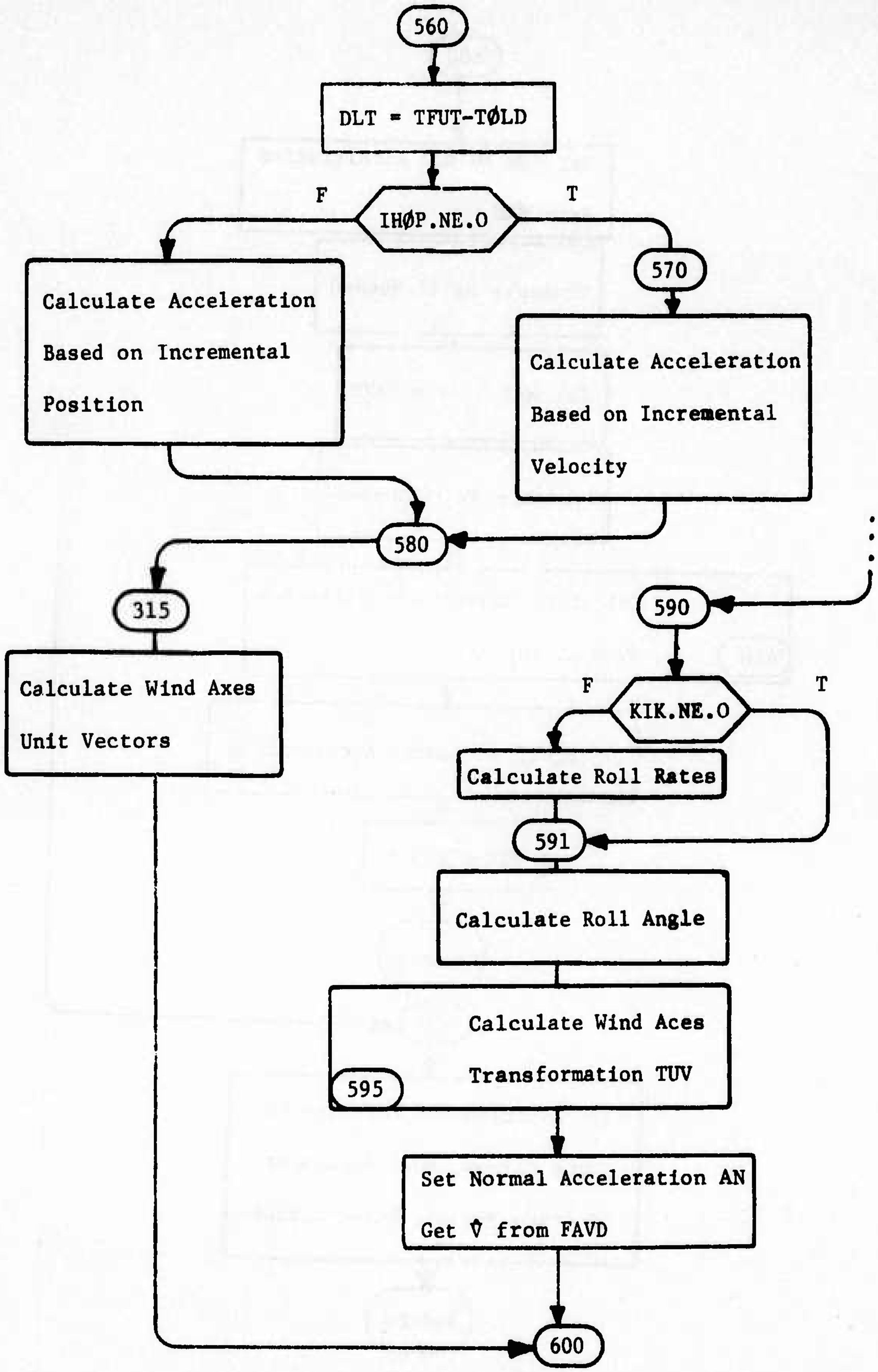

Figure 4. Flow Diagram of ESCAPE (Continued)

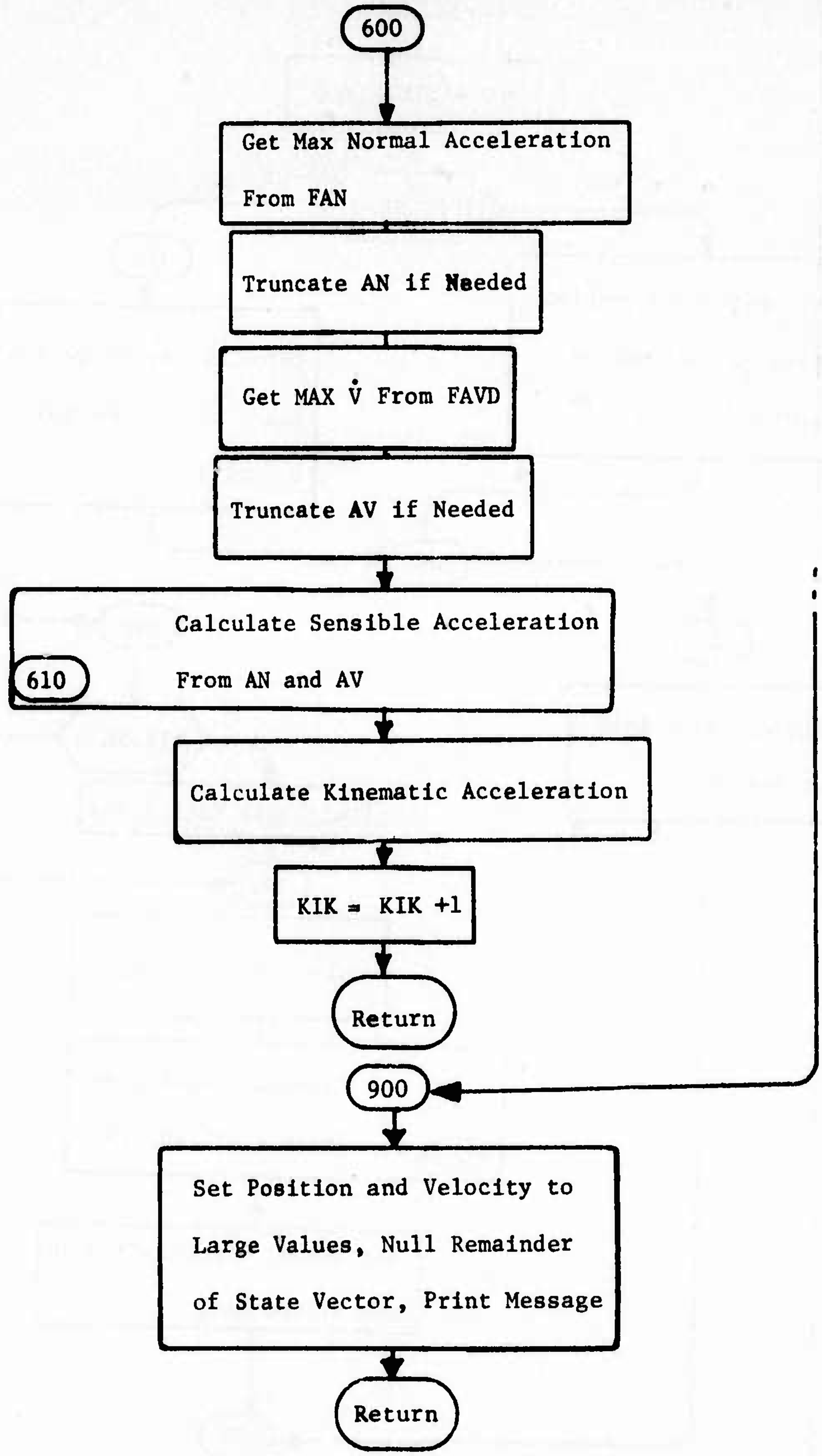

Figure 4. Flow Diagram of ESCAPE (Concluded)

$$X(t) = X(t_o) + A_K(t_o)\frac{(t-t_o)^2}{2} \qquad (25)$$

In use, the maximum value that $t-t_o$ will normally attain is 0.1 second. $A_K(t_o)$ is repeatedly updated as the simulation proceeds and, for instance, when IZG = 1, a feedback loop is established and the acceleration $A_K(t_o)$ and time t_o are continually being updated based on the present position of the target. Figure 4 is a flow diagram of the logic of ESCAPE.

INPUT DATA TO DEFINE A MANEUVER PROGRAM

Card No.	Format	Data
1.	A6, I4	Card title, number of maneuvers in motion program. Maximum number of maneuvers is 10.
2.	A6, I4, 5E10.2	Card title, maneuvers index, maneuver parameters. Maneuver parameters vary with index.
3.	A6, I4, 4(I5,E10.2) /(10x4(I5,E10.2))	Card title, number of lower limits, sequence of (state vector index, lower limit value) pairs. State vector indices are defined below. Maximum number of limits is 10.
4.	A6, I4, 4(I5, E10.2) /(10x4(I5,E10.2))	Card title, number of upper limits, sequence of (state vector index, upper limit value) pairs. State vector indices are defined below. Maximum number of limits is 10.

Cards 2, 3, and 4 are repeated in sequence for each maneuver in the program.

Card No.	Format	Data
5.	A6, I4, 6E10.2	Card title, starting maneuver index less 1, initial conditions of target, x_1, x_2, x_3, V_1, V_2, V_3. The simulation uses X_3 down.

Card sequence [1, (2, 3, 4), (2, 3, 4), -- -- (2, 3, 4)] may be repeated for additional runs of the target. The minimum number of cards in a data sequence is 5.

MANEUVER PARAMETERS

There are five maneuver parameters read from each No. 2 card, and

the role played by each parameter depends upon the maneuver being defined. In all cases parameter 5 is the throttle setting. Negative values for IZG indicate that the parameters are defined in the target wind coordinate system.

IZG = 1, or -1	Parameter 1, 2, and 3 designate a point in space x_1, x_2, and x_3 toward which the target will fly. Parameter 4 is the time allowed for the maneuver.
IZG = 2, or -2	Parameters 1, 2, and 3 locate a point in space an increment Δx_1, Δx_2, Δx_3 away from the initialization of this maneuver. The target will try to fly to this point in the amount of time given by parameter 4.
IZG = 3, or -3	Parameters 1, 2, and 3 locate a moving point that in the limit leads the target parallel to Δx_1, Δx_2, Δx_3 specified by the parameters. Again parameter 4 is the time allotted to perform the maneuver. In effect, a constant acceleration is specified.
IZG = 4, or -4	Parameters 1, 2, and 3 define a velocity vector that the target will attempt to achieve in parameter 4 number of seconds.
IZG = 5, or -5	Parameters 1, 2, and 3 define an incremental velocity vector that will be added to the present velocity over parameter 4 seconds.
IZG = 6, or -6	Parameters 1, 2, and 3 define an incremental velocity that is used with parameter 4 to define a constant acceleration. This case is essentially the same as IZG = 3 except velocity rather than displacement is used to define acceleration.
IZG = 7	Parameter 1 is the initial roll angle in degrees. Parameter 2 is the roll rate in degrees if parameter 4 is zero. If parameter 4 is not zero, the roll rate is computed as (parameter 2 minus parameter 1)/parameter 4. Parameter 3 is the specified normal acceleration in g's. Tangential acceleration is controlled by parameter 5 as a throttle setting. Parameter 5 has values between 0. and 1.0.

STATE VECTOR INDICES

As described in the data and programming subsection, the target motion program steps up to the next maneuver in the sequence whenever a lower or upper limit is violated. A maximum of 10 upper and 10 lower limits is allowed per maneuver. At least one upper and one lower limit must be specified. The state vector quantity desired as a maneuver trigger is specified by state vector component index. The following list of indices must be used when formulating the limit control cards:

State Vector Quantity	Index
Simulation time	1
Attacker x_1 position	2
Attacker x_2 position	3
Attacker x_3 position	4
Target x_1 position	32
Target x_2 position	33
Target x_3 position	34
Range, attacker to target	31
Attacker velocity V_1	5
Attacker velocity V_2	6
Attacker velocity V_3	7
Attacker speed	8
Target velocity V_1	35
Target velocity V_2	36
Target velocity V_3	37
Target speed	38
Closing rate, attacker to target	56
Attacker roll angle ϕ	9
Attacker pitch angle θ	10
Attacker heading angle Ψ	11

Target roll angle ϕ	46
Target pitch angle θ	45
Target heading angle Ψ	44
Angle off, attacker to target	57
Target normal acceleration (lift)	42
Target tangential acceleration (drag), (thrust)	43

ADDITIONAL DATA

In addition to the target motion program control data described above, the target simulation requires tables of performance data. Specifically, four tables of data are required so that the aerodynamic performance of the target may be realistically constrained.

Maximum Normal Acceleration A two-dimensional table of maximum normal acceleration in feet/second^2 versus mach number and altitude is required. This table gives the maximum available normal acceleration as a function of the target's mach number and altitutde. It is used to make sure that the target does not outperform its capabilities. This table is described by the ordinate data arranged to vary with mach number most rapidly (down the columns) and the two abscissa arrays, mach number and altitude in feet. Formats for all are (10x7E10.2), with read order of mach vector, altitude vector, and then table.

Maximum Thrust To simulate the speed characteristics of the target properly, a table of maximum available thrust versus mach number and altitude is needed. Thrust in feet/second^2 is used with throttle position to control the tangential $\dot{V}$, acceleration of the target. This table is also two-dimensional with mach number as the column variable and altitude as the row variable. The formats are also (10x,7E10.2), with reading in the order mach vector, altitude vector, then table.

Drag Drag coefficient as a function of normal acceleration, mach number, and altitude is required in a three-dimensional format. The table defines a volume with each vertical slice defining a two-dimensional table. In the two-dimensional table the data varies down the columns with normal acceleration and from column to column with mach number. Each two-dimensional table is associated with an altitude. The read details are a format

of (10x7E10.2) and read sequence is normal acceleration vector, mach vector, altitude vector, and then table.

Angle of Attack The target simulation does not need the angle of attack for aerodynamic purposes. However, the actual attitude of the target with respect to the wind axes may affect the evaluation of a gun system. For this reason the target angle of attack is used to complete the attitudinal picture. For this purpose, angle of attack is considered a function of normal acceleration, mach number, and altitude. A three-dimensional table is used with column-wise organization with respect to varying normal acceleration, row-wise organization with respect to increasing mach number, and slice-wise organization with respect to increasing altitude.

ANCILLARY PROGRAMS

The target simulation uses three ancillary programs to do the data reading and table look up for the normal acceleration, thrust, drag and angle of attack. These programs are described briefly in the following paragraphs.

Function FAN handles the data manipulation used to describe the maximum normal acceleration available as a function of mach number and altitude. Upon the first call to FAN, data describing a table of normal acceleration versus mach number and altitude is read and stored. In the read sequence, the reference values for the mach number are read first and stored in array AM in ascending order according to magnitude. Next the reference values for the altitude in feet are read and stored in the array AH according to ascending index and magnitude. Finally the tabular values for normal acceleration in feet per second are read and stored in FMH. FMH is assumed to be structured as a two-dimensional array with the column entries indexed to the mach number array and the row entries indexed to the altitude array. Each data card has the first ten columns reserved for a label which is read and pointed via LABEL. The presently implemented program has three data entries in AM and AH and nine entries in FMH. Variables NM and NH indicate that the data is 3 X 3.

On calls subsequent to the initial call, a value for the normal acceleration, FAN, is calculated by the two variable table look-up program LOOK2 using VM and H as input values for mach number and altitude.

Function FAVD controls the data and calculations needed to obtain the maximum available tangential, or V, acceleration. On the first call to FAVD data is read and stored that defines two tables: One for maximum available thrust as a function of mach number and altitude, and one for drag-to-lift ratio as a function of normal acceleration, mach number and altitude.

The read sequence first reads the reference values of normal acceleration in ft $/S^2$ and stores them in the array AAN. Next reference values for the mach number are read and stored in array AM. The third read fills the array AH with the reference values of the altitude in feet. In the fourth read the tabular values of the maximum available thrust are read into array TRST in a two-dimensional array format. Row-wise organization is indexed according to the reference values stored in AM, the mach number, and column-wise organization is indexed according to the values stored in AH, altitude.

The fifth read ingests and stores the table of drag ratios into array DRG. These are nondimensional ratios of the drag to lift at the various normal acceleration (lift), mach number, and altitude reference points stored in AAN, AM, and AH, respectively. The array DRG varies most rapidly with normal acceleration (row-wise organization), next most rapidly with mach number (column-wise organization) and least rapidly with altitude (slice-wise variation). DRG is structured as a three-dimensional array with first subscript associated with altitude. Again the first 10 columns of the data cards are reserved for LABEL to be used strictly as a label.

On subsequent calls to FAVID, AN, VM and H are used by LOOK3 to look up and interpolate a value for DRAG, and VM and H are used by LOOK2 to look up and interpolate a value for THRST. The $\dot{V}$ acceleration for the target is calculated as

$$FAVD = THRST * TP - DRAG * AN$$

where $0. \leq TP \leq 1.$ and TP is the throttle position variable.

NA, NM, and NH show that the implemented program expects a 3 X 3 TRST and a 3 X 3 X 3 DRG.

<u>Function TAAF</u> is used to read and control the data for the definition of the target angle of attack. The angle of attack is considered as a function of three variables: normal acceleration, mach number, and altitude.

Upon the first call to TAAF the following read and store sequence is activated: first reference values for target normal acceleration in ft $/S^2$ are read and stored in array TAN; second reference values for mach number are read and stored in array TM; third reference values for altitude in feet are read and stored in array TH; and finally the three-dimensional table of angle of attack in degrees is read and stored in TAA. The assumed organization of TAA is that of a three-subscripted array with the first index linked to the values in TAN, the second index linked to the values in TM, and the third index linked to the reference values in TH. Each data card has the first 10 columns reserved for a label which is read and printed from LABEL. The expected data organization for the implemented program is 3 X 3 X 3 as indicated by NA, NM and NH.

On subsequent calls LOOK3 uses normal acceleration, AN, mach number, VM, and altitude, H, to look up and interpolate a value for angle of attack TAAF.

SECTION IV

SIGHT SYSTEM

SIGHT MODES

Three basic sight modes were adapted for implementation in the lead pursuit model: (1) A perfect first order prediction using the basic methodology present in GAMES (Reference 1); (2) a second order modification of the linear prediction model; and (3) an existing sight simulation. Modes (1) and (2) are perfect systems in that they assume position, velocity, and accelerations of the target are all known and available. Mode (3) is the disturbed-reticle sight supplied by AFATL personnel (AFAL-TR-75-52).

SIGHT ALGORITHM 1

In the first algorithm, the perfect first order prediction, the time of flight, and range are determined using the iterative procedure described in GAMES (Reference 1). The relative position, relative velocity, and acceleration vectors in the inertial system are calculated from

$$\begin{aligned} P_{rel} &= P_{targ} - P_{att} \\ V_{rel} &= V_{targ} - V_{att} \end{aligned} \tag{26}$$

The initializing range is taken as the present range from attacker to target.

The projectile average velocity is calculated as was done in GAMES using the appropriate projectile drag coefficients from subroutine DRAG (in GAMES). From these, a time of flight is calculated and used to calculate a new range. This procedure is repeated until the change in range is sufficiently small. The final range and time of flight values are used with attacker position and velocity vectors and gun information (muzzle velocity and position) to calculate the future position of the projectile one time of flight in the future. The proper pipper position is then calculated by backing up from the projectile future position an amount and in a direction determined from the target velocity and position.

SIGHT ALGORITHM 2

The second order approximation is accomplished by the addition of acceleration terms during calculation of both target and attacker positions after one time of flight. Otherwise, the algorithm is exactly the same as algorithm 1. In fact, the second order algorithm is imbedded in the linear prediction model and controlled by a flag.

The entire first and second order algorithm was extracted from GAMES, modified and installed in a separate subroutine called SSIGHT. Information describing the gun is transmitted through common block GUN while encounter data is transmitted through common block STATE.

SIGHT ALGORITHM 3

Subroutine SIGHT contains the disturbed-reticle sight algorithm. This routine was taken from technical report AFAL-TR-75-52 (Reference 4) and modified to accept encounter information externally through common STATE. The algorithm presented in Reference 4 assumed a particular encounter geometry and contained a loop which carried out the encounter. The modified routine uses the attacker information available in STATE, and makes one pass using the lead angles calculated from the previous pass through the algorithm and stored. The routine does not utilize the target information available in STATE. Thus, the algorithm assumes no target information is available and positions the pipper to indicate the position in the HUD of the impact point of a projectile fired one time of flight prior.

DATA REQUIREMENTS

The general data requirements are that, in the event that sight parameters are not fixed, they will be read in upon the first call to the sight program. This means that the subprogram writer will have to include logic in the sight program to recognize the first call. Further, some sight systems may require some standard initialization whenever a new case is begun. This report contains three implementations that can serve as examples for the programmer.

The first order sight requires a minimum amount of data. The gun muzzle velocity and the gun angle are the only data needed for the first order sight.

The second order sight is imbedded within the same program as the first order sight and consequently its data is taken care of by that program.

The disturbed reticle sight program SGHT is an example of a mechanization of a sight system. It is included as the third option for this attacker-target simulation. For this program, the program listing provides the best display of the needed information. For a different sight, this program would be rewritten and subroutine SGHT might read different data entirely, although there would be obvious similarities.

DATA FOR THE SIGHT PROGRAMS

Upon the first call to SSIGHT in the initialization phase of each new engagement, data defining the sight system is read and stored. There is one

READ command using the FORMAT (A6,I4,7E10.2). The following quantities are read from one card in the card in the card input file:

LABEL	a six character identifier.
IORD	order of sight system. IORD = 1 = first order perfect. IORD = 2 = second order perfect. IORD = 3 = sight implemented in subprogram SGHT.
VELMUZ	muzzle velocity, feet per second
GUNALP	gun angle, degrees.
GHR	gun harmonization range, feet.
DA	location of gun in azimuth direction from HUD, feet.
DE	location of gun in elevation direction from HUD, feet.
HALP	HUD angular displacement from airframe longitudinal axis, degrees.
SIG	Sight system damping factor.

SECTION V

IMPLEMENTATION INTO GAMES

GAMES

The Joint Technical Coordinating Group (JTCG) guns study program GAMES is essentially a design program for aerial gunnery systems whereby the effects of gun system parameters can be studied. The effective procedure of using the program is to set the gun system parameters and then evaluate their effectiveness by running a spectrum of engagements. The relevance of this concept to the present effort is that provision had to be made for a sequence of initial conditions and engagement descriptors to be read in by the simulation.

STRUCTURE OF GAMES

The structure of the GAMES program was not violated by the present effort. In fact, the program resulting from the inclusion of aerodynamic lead pursuit into GAMES still has all of the features and capabilities of the original version. This was accomplished by taking advantage of the fact that GAMES was written to be able to receive flight path data for both target and attacker from an outside source. The present attacker and target modelling program simply becomes another outside source of data. In this manner, GAMES itself was only modified to the extent of a third read option, a call to PILOT, and a few branches to take advantage of the fact that the new simulation produces more data than before.

PILOT

Subroutine PILOT has been explained in some detail in Section II of this report, especially with respect to the control algorithm aspects. PILOT as a subprogram also has executive duties. It must make appropriate initiation reads when a new case is started as well as initiation calls to TURKEY ESCAPE and SSIGHT. It also must envoke ESCAPE and SSIGHT after an integration of the equation of motion has been made and a reference state reached.

As indicated above, reinitialization is needed each time GAMES goes on to its next case. This is accomplished by calling PILOT with time set to zero. Upon this call PILOT calls the other programs in the attacker-target simulation with the initialization flag set, as well as reading its own initialization data.

For regular calls, time greater than zero, the PILOT program is first called and after integration of the simulation by PILOT, a call to ESCAPE is with the reference point flag set.

Subroutine PILOT is a slave to the main program of GAMES and does not have the capability of terminating the simulation. A listing of the main program of GAMES is included in the appendix along with PILOT and the other programs used in the simulation.

CDC 6600 RUNS

The modified GAMES program was implemented on the Eglin AFB CDC 6600 digital computer system. Test cases using engagements and data supplied by DLYD were run. Sample output from the simulation is included in the appendix to this report.

SECTION VI

OVERVIEW OF INPUT FILE STRUCTURE

INITIAL CALL TO PILOT

Upon the first call to PILOT with zero argument, a sequence of reads is initiated. The structure of the input file must be consistent with this sequence of reads. The basic initial sequence is as follows.

PILOT

Integration and print-out parameters and attacker initial conditions as described in Section II. Requires 4 cards.

ESCAPE

Data as described in detail in Section III. Upon the initialization of ESCAPE as now written a minimum of 25 cards will be read.

SSIGHT

Each time PILOT is called with zero argument SSIGHT will read one card as described in Section IV.

TURKEY

Upon the initial call to PILOT, all of the tabular data described in the IMAGE (Reference 1) program will be read. The card file must contain the following data sequences:

IDUM1	Integer data totaling 7 cards.
TDUM1	Real data totaling 69 cards.
TDUM2	Real data totaling 588 cards.
DUM	Real data totaling 4 cards.

Additions to DUM totaling 1 card

Total Cards on Initial Read

The above described card files give a minimum card set of 699 cards to be read as the initial data.

SUBSEQUENT RE-RUNS

Should reinitialization of the simulation be required, a call to PILOT with zero for argument will cause reinitialization of the programs. A much reduced data file is required as follows:

PILOT	3 cards, initial conditions only.
ESCAPE	2 cards minimum (see Section III).
SSIGHT	1 card
TURKEY	1 card, update on DUM only.

REFERENCES

1. Solomon, G., Caluda, M. J., Gunnery Analysis, Modular Effectiveness Simulation (GAMES) AFATL-TR-75-118, September, 1975.

2. Berger, J. B., M. Meyers, R. R. Wallace, Improved Model for Aerial Gunnery Effectiveness. AFATL-TR-68-112, September, 1968.

3. Whitehouse, G. D., A. J. McPhate, L. Theriot, A Research and Analysis Program of Studies on the Analysis of Weapon Effectiveness, Phase II Results, AFATL-TR-71-110, August, 1971.

4. Manske, Robert A., Air to Air Gunfire Control Equations for Digital Lead Computing Optical Sights, AFAL-TR-75-52, January, 1975.

5. McDonnel-Douglas Corporation, IMAGE User's Manual. Supplement to AFATL-TR-68-112, September, 1968.

APPENDIX A

PROGRAM LISTINGS AND SAMPLE RUN

```
      SUBROUTINE PILOT( TNEW )
      REAL  LKSLZR
      COMMON /GUN/ GUNS(5)
      COMMON /SIGHTS/ALPHUD,SGTSIG
      COMMON /FROGS/ TWB(3,3),TWE(3,3),TWBD(3,3),W(3,3),WD(3,3)
      COMMON / KILLER / GOTCHA
      LOGICAL FLAG1,FLAG2,FLAG3
      LOGICAL KONOFF(4)
      INTEGER JB(2) , JY(2)
      COMMON/PLOP/NO,BV(456),YV(456),RV(456),TV(456)
      DIMENSION ZONK(456,4)
      EQUIVALENCE(ZONK(1,1),BV(1)),(ZONK(1,2),YV(1)),(ZONK(1,3),RV(1))
     & , (ZONK(1,4),TV(1))
      DIMENSION TD(3),XDD(3),YDD(3),DDX(3)
      DIMENSION Y(3),YD(3),        TE(3),D(3)
     P               ,SVDS(15),ST(15)
      DIMENSION X(3),XD(3),ZERO(4)
      DIMENSION DX(3)
      DIMENSION EV(3)
      DIMENSION BOOGIE(7,8)
     &               ,TBE(3,3)
       DIMENSION ARRDDS(2)
      DIMENSION           CVCTR(4),ERVCTR(3),WS(3,3)
      DIMENSION EPS1(4),EPS2(4),EPSD1(4),EPSD2(4)
      COMMON /PILOTS/ BOOGIE,D,DX
      COMMON /STATE/ TIME,SV(30),TSV(30),SVD(15),XHUD,YHUD,XHUDD,YHUDD
      EQUIVALENCE ( SV(26),CVCTR(1)),(SYANK,SINY),(CYANK,COSY)
      EQUIVALENCE ( RDOT,RANGED )
      EQUIVALENCE( EV(1),ERVCTR(1) ),( R,RANGE )
     &               ,(RANGE,SV(30))
     &               ,(TBE(1,1),SV(16))
      EQUIVALENCE ( YANKD,YDOT )
     &               ,(Y(1),TSV(1)),(YD(1),TSV(4)),(YDD(1),TSV(8))
     &     ,(X(1),SV(1)),(XD(1),SV(4)),(XDD(1),SVD(4))
     &               ,(SV(11),ALPHA),(SVD(11),ALPHAD)
      EQUIVALENCE( JB(1),JBP ),( JB(2),JBN ),( JY(1),JYP ),( JY(2),JYN )
      EQUIVALENCE(BOOGIE(7,1),BANK),(BOOGIE(7,2),BANKD),
     1(BOOGIE(7,3),YANK),(BOOGIE(7,4),YANKD),(BOOGIE(7,5),RDIPH),
     2(BOOGIE(7,6),RDOT),(BOOGIE(7,7),SIGMA),(BOOGIE(7,8),SIGMAD)
      EQUIVALENCE ( ARRDDS(1),ARRDDP ),( ARRDDS(2),ARRDDN )
```

```
      KWAK(K) = K - ( K / 2 ) * 2 + 1                                   PBGN 410
      DATA EPSRC/0.05/                                                  PBGN 420
      DATA EPSAR / 0.05 /                                               PBGN 430
      DATA EPSARD / 0.05 /                                              PBGN 440
      DATA JUMP/0/                                                      PBGN 450
      DATA EPS1 / 0.05 , 0.05 , 100.0 , 0.05 /                          PBGN 460
      DATA EPS2 / 0.01 , 0.01 , 200.0 , 0.01 /                          PBGN 470
      DATA EPSD1 / 0.10 , 0.10 , 50.0 , 0.10 /                          PBGN 480
      DATA EPSD2 / 0.05 , 0.05 , 100.0 , 0.05 /                         PBGN 490
      DATA PIE2 /1.5708 /                                               PBGN 500
      DATA WGHT1,WGHT2 / 0.8992 , 0.1008 /                              PBGN 510
      DATA DUCK1,DUCK2,DUCK3/0.3980,0.2506,0.3107/                      PBGN 520
      DATA LKSLZ9 / 0.5312 /                                            PBGN 530
       DATA PIE / 3.14159 /                                             PBGN 540
      IF( TNEW.GT.0.0 ) GOTO 4000                                       PBGN 550
      IF( JUMP.GT.00 ) GOTO 4100                                        PBGN 560
      JUMP = 01                                                         PBGN 570
      READ 3,LABEL,NF1,NF9,DTIME                                        PBGN 580
      PRINT7,LABEL,NF1,NF9,DTIME                                        PBGN 590
      DTIMEP = DTIME * 1.010                                            PBGN 600
      DTIMEM = DTIME * 0.010                                            PBGN 610
      FLAG2 = .FALSE.                                                   PBGN 620
      IF( NF9.GT.00 ) GOTO 4103                                         PBGN 630
      FLAG2 = .TRUE.                                                    PBGN 640
      NF9 = -NF9                                                        PBGN 650
 4103  CONTINUE                                                         PBGN 660
 4100 READ4,LABEL,(SV(I),I=1,15)                                        PBGN 670
      SV(7) = SQRT( SV(4)**2+SV(5)**2+SV(6)**2 )                        PBGN 680
      PRINT 8,LABEL,(SV(I),I=1,15)                                      PBGN 690
      DO 4101 I=1,4                                                     PBGN 700
      ZERO(I) = 0.0                                                     PBGN 710
 4101 CVCTR(I) = 0.0                                                    PBGN 720
      TIME = 0.0                                                        PBGN 730
      CALL ESCAPE(-1)                                                   PBGN 740
      RANGE = 0.0                                                       PBGN 750
      DO    4102   J=1,3                                                PBGN 760
 4102  RANGE = RANGE + ( Y(J)-X(J) )**2                                 PBGN 770
      RANGE = SQRT( RANGE )                                             PBGN 780
      CALL SSIGHT(-1)                                                   PBGN 790
      CALL TURKEY(-1)                                                   PBGN 800
```

```
      DO          3001          J=1.4
3001  BOUGIE(1.J*2-1) = CVCTR(J)
      CALL TURKEY(-2)
      DO          3002          J=1.4
      BOUGIE(1.J*2) = CVCTR(J)
3002  CVCTR(J)=0.0
      RETURN
C     CVCTR( 1 ) = DA
C     CVCTR(2) = DDS
C     CVCTR(3) = CDHM
C     CVCTR(4) = DR
4000  KK = NF1-01
      KKK = NF9-01
      AICH = DTIME
      AICH2 = 0.5*AICH
400   IF( TIME.LE.TNEW-DTIMEP ) GOTO 4200
      IF( TIME.GE.TNEW-DTIMEM ) GOTO 4700
      AICH = TNEW-TIME
      AICH2 = 0.5*AICH
4200  KK = KK +01
      IF( KK.NE.NF1 ) GOTO 4300
      KK = 00
      DO          4201          J=1.4
      IF( J.EQ.04 ) GOTO 4201
      ZERO(J) = WGHT1*ZERO(J) + WGHT2*CVCTR(J)
4201   CVCTR(J) = ZERO(J)
      CALL ESCAPE(1)
      CALL TURKEY(1)
      DO 1001  J = 1.3
        DX(J) = YD(J) - XD(J)
1001  D(J) = Y(J) - X(J)
      RANGE = D(1)*D(1) + D(2)*D(2) + D(3)*D(3)
      RSQ = RANGE
      RANGE = SQRT( RANGE )
      RST = SQRT( R )
      RDIFH = R - GUNS(3)
      DO 2002  I = 1 , 3
      SUM = 0.0
      DO 2001  J = 1 , 3
2001  SUM = SUM   +  IBE( J.I ) * D(J)
```

```
      ERVCTR(I) = SUM                                                   PBGN1210
 2002 CONTINUE                                                          PBGN1220
C  NOTE THAT THE VEL AND THE ACC OF THE PIPPER IS ASSUMED TO BE APPROX 0PBGN1230
      DO 561 I = 1 , 3                                                  PBGN1240
      SUM = 0.0                                                         PBGN1250
      DO 560 K = 1 , 3                                                  PBGN1260
  560 SUM = SUM + W(I,K) * EV(K)                                        PBGN1270
  561 TE(I) = SUM                                                       PBGN1280
      DO 565 I = 1 , 3                                                  PBGN1290
      SUM = 0.0                                                         PBGN1300
      DO 564 K = 1 , 3                                                  PBGN1310
  564 SUM = SUM + TBE(K,I) *    DX(K)                                   PBGN1320
  565 TE(I) = SUM - TE(I)                                               PBGN1330
      RDOT = TE(1)*EV(1)+TE(3)*EV(3)+TE(2)*EV(2)                        PBGN1340
      RDOT = RDOT / R                                                   PBGN1350
  405 CONTINUE                                                          PBGN1360
      CALL SSIGHT( 1 )                                                  PBGN1370
      IF( EV(3) .EQ. 0.0 ) EV(3) = 1.0E-06                              PBGN1380
      IF( EV(2) .EQ. 0.0 ) EV(2) = 1.0E-06                              PBGN1390
      ERS3 = EV(3) * EV(3)                                              PBGN1400
      ERS2 = EV(2) * EV(2)                                              PBGN1410
      ERS1 = EV(1)* EV(1)                                               PBGN1420
      VIVZ2 = ERS1 + ERS2                                               PBGN1430
      SINY = - ERVCTR( 3 ) / RANGE                                      PBGN1440
      CYANK2 = 1.0 - SYANK*SYANK                                        PBGN1450
      CYANK = SQRT( CYANK2 )                                            PBGN1460
      SYANKD = ( - TE(3)*R + EV(3)*RDOT ) / RSQ                         PBGN1470
      YANK = ARSIN(SYANK )  - ALPHUD - YHUD                             PBGN1480
      YANKD = SYANKD / CYANK - YHUDD                                    PBGN1490
      SYANK = SYANK * YANKD * SYANKD                                    PBGN1500
      SYANKD = 2.0 * R * RDOT * SYANKD                                  PBGN1510
      SKSQR = SQRT( VIVZ2 )                                             PBGN1520
      SINS = ERVCTR(2)/SKSQR                                            PBGN1530
      COSS = ERVCTR(1)/SKSQR                                            PBGN1540
      TSV(25) = RDOT                                                    PBGN1550
      SIGMA = ATAN2( SINS,COSS )                                        PBGN1560
      TSV(26) = SIGMA                                                   PBGN1570
      SIGMA = SIGMA - XHUD                                              PBGN1580
      SIGMAD = (TE(2) - RDOT*COSY*SINS + YDOT*R*SINY*SINS)/(R*COSY*COSS)PBGN1590
     P - XHUDD                                                          PBGN1600
```

BEST AVAILABLE COPY

```
      EVTE = 2.0*( EV(1)*TE(1) + EV(2)*TE(2) ) * SIGMAD              PBGN1610
      YKSG2 = YANK*YANK + SIGMA*SIGMA                                   PBGN1620
      ERR = SQRT( YKSG2 )                                               PBGN1630
      YNKP = YANK + YHUD + ALPHUD + LKSLZR                              PBGN1640
      BANK = ATAN2( SIGMA,YNKP )                                        PBGN1650
      YPSG2 = YNKP*YNKP + SIGMA*SIGMA                                   PBGN1660
      BANKD = ( YNKP*SIGMAD - YANKD*SIGMA )/YPSG2                       PBGN1670
      YPSG2D = 2.*BANKD*( YANKD*YNKP+SIGMA*SIGMAD )/YPSG2               PBGN1680
      FLAG3 = ERR.LE.EPSBC                                              PBGN1690
      ERRD = ( YANK * YANKD + SIGMA * SIGMAD ) / ERR                    PBGN1700
      L = 01                                                            PBGN1710
  403 CONTINUE                                                          PBGN1720
      DO 410  I = 1 , 4                                                 PBGN1730
  410   CVCTR(I) = ZERO(I)                                              PBGN1740
      JOY = ( L + 1 ) / 2                                               PBGN1750
      CVCTR( JOY ) = BOOGIE( 1,L )                                      PBGN1760
      ASSIGN 447 TO KRASH                                               PBGN1770
  411 CONTINUE                                                          PBGN1780
      CALL TURKEY(2)                                                    PBGN1790
  402 CONTINUE                                                          PBGN1800
      DO 551  I = 1 , 3                                                 PBGN1810
      DDX(I) = YDD(I) - XDD(I)                                          PBGN1820
      DO 551  J = 1 , 3                                                 PBGN1830
      SUM= 0.0                                                          PBGN1840
      DO 550  K = 1 , 3                                                 PBGN1850
  550 SUM = SUM + W(I,K) * W(K,J)                                       PBGN1860
  551 WS(I,J) = SUM + WD(I,J)                                           PBGN1870
      DO  553  I = 1,3                                                  PBGN1880
      SUM = 0.0                                                         PBGN1890
      SUM1 = 0.0                                                        PBGN1900
      DO 552 K = 1,3                                                    PBGN1910
      SUM = SUM + WS(I,K) * EV(K)                                       PBGN1920
      SUM1 = SUM1 +  W(I,K) * TE(K)                                     PBGN1930
  552 CONTINUE                                                          PBGN1940
  553 TD(I) = SUM + SUM1 + SUM1                                         PBGN1950
      DO 558  I = 1 , 3                                                 PBGN1960
      SUM = 0.0                                                         PBGN1970
      DO 557  K = 1 , 3                                                 PBGN1980
  557 SUM = SUM + TBE(K,I) *  DDX(K)                                    PBGN1990
  558 TD(I) = SUM - TD(I)                                               PBGN2000
```

```
      SUM = EV(1)*TD(1)+EV(2)*TD(2)+EV(3)*TD(3)                           PBGN2010
      SUM1 = TE(1)*TE(1)+TE(2)*TE(2)+TE(3)*TE(3) + SUM                    PBGN2020
       RDDOT = (SUM1 - RDOT * RDOT )/ R                                   PBGN2030
      SIGMDD = ( TD(2)*TE(1)+TE(2)*TD(1) - EVTE ) / VIVZ2                 PBGN2040
      YANKDD = ( - TD(3)*R + EV(3)*RDDOT - SYANKD ) / RSQ                 PBGN2050
      YANKDD = ( YANKDD * CYANK + SYANK ) / CYANK2                        PBGN2050
      BANKDD = ( YNKP*SIGMDD-YANKDD*SIGMA )/YPSG2 - YPSG2D                PBGN2070
      ERRDD = (YANK*YANKDD+SIGMA*SIGMDD+YANKD**2+SIGMAD**2-ERRD**2)/ERR   PBGN2080
       GOTO KRASH,(446,447)                                               PBGN2090
  447  CONTINUE                                                           PBGN2100
      BOOGIE(2,L) = ERRDD                                                 PBGN2110
      BOOGIE(3,L) = BANKDD                                                PBGN2120
      BOOGIE(4,L) = YANKDD                                                PBGN2130
      BOOGIE(5,L) = RDDOT                                                 PBGN2140
      BOOGIE(6,L) = SIGMDD                                                PBGN2150
      L = L + 01                                                          PBGN2160
      IF ( L .LE. 08 ) GOTO 403                                           PBGN2170
       KIK = 04                                                           PBGN2180
      IF( FLAG3 ) KIK = 01                                                PBGN2190
      DO  420  I = 1 , 7 , 2                                              PBGN2200
      IP1 = I + 01                                                        PBGN2210
      JOY = IP1 / 2                                                       PBGN2220
      J = JOY + 02                                                        PBGN2230
      IF( JOY .EQ. KIK ) GOTO 851                                         PBGN2240
      ARR = BOOGIE(7,I)                                                   PBGN2250
      ARRD = BOOGIE(7,IP1)                                                PBGN2260
      JBP = I                                                             PBGN2270
      JBN = IP1                                                           PBGN2280
      ARRDDP = BOOGIE( J,I )                                              PBGN2290
      ARRDDN = BOOGIE( J,IP1 )                                            PBGN2300
      IF ( ARRDDP .GT. ARRDDN ) GOTO 412                                  PBGN2310
      JBP = IP1                                                           PBGN2320
      JBN = I                                                             PBGN2330
      ARRDDP = BOOGIE( J,IP1 )                                            PBGN2340
      ARRDDN = BOOGIE( J,I )                                              PBGN2350
  412  CONTINUE                                                           PBGN2360
C  K = 01  MOST POSITIVE  -- JBP                                          PBGN2370
C  K = 02  MOST NEGATIVE  -- JBN                                          PBGN2380
      K = 01                                                              PBGN2390
      IF ( ABS(ARR) .GT. EPSAR ) GOTO 805                                 PBGN2400
```

```
      IF( ABS ( ARRD ) .LT. EPSARD ) GOTO 851                      PBGN2410
  805 CONTINUE                                                       PBGN2420
      IF ( ARR .EQ. 0.0 ) GOTO 820                                   PBGN2430
      IF ( ARR * ARRD .GE. 0.0 )  GOTO 821                           PBGN2440
  810 T = - ARRD / ARRDDS ( K )                                      PBGN2450
      IF ( T .GT. 0.0 ) GOTO 815                                     PBGN2460
      K = KWAK( K )                                                  PBGN2470
      IF ( ARRDDS( 1 )*ARRDDS( 2 ) .GT. 0.0 ) GOTO 850               PBGN2480
      GOTO 810                                                       PBGN2490
  815  GOOFUS = ARR + ARRD * T + ARRDDS(K) * T * T * 0.50            PBGN2500
      GOOFUS = GOOFUS * SIGN( 1.0,ARR )                              PBGN2510
      IF ( GOOFUS .GT. EPSAR ) GOTO 821                              PBGN2520
      GOTO 850                                                       PBGN2530
  820 IF ( ARRD * ARRDDS(K) .GT. 0.0 ) GOTO 825                      PBGN2540
      GOTO 850                                                       PBGN2550
  821  IF ( ARR * ARRDDS(K) .GT. 0.0 )GOTO  825                      PBGN2560
      GOTO 850                                                       PBGN2570
  825  CONTINUE                                                      PBGN2580
      K = KWAK( K )                                                  PBGN2590
      GOTO 850                                                       PBGN2600
  851  CVCTR(JOY) = ZERO(JOY)                                        PBGN2610
      KONOFF( JOY ) = .TRUE.                                         PBGN2620
      GOTO 420                                                       PBGN2630
  850 CONTINUE                                                       PBGN2640
      CVCTR( JOY ) = BOOGIE( 1,JB( K ) )                             PBGN2650
      KONOFF( JOY ) = .FALSE.                                        PBGN2660
  420 CONTINUE                                                       PBGN2670
      DO          4400        J=1,4                                  PBGN2680
      IF(  KONOFF(J)  )   GOTO  4400                                 PBGN2690
      BOOG = ABS( BOOGIE(7,2*J-1) )                                  PBGN2700
      BOOGD = ABS( BOOGIE(7,2*J) )                                   PBGN2710
      IF( BOOG.LT.EPS1(J).AND.BOOGD.LT.EPSD1(J) )                    PBGN2720
     &            CVCTR(J) = CVCTR(J)*0.5                            PBGN2730
      IF( BOOG.LT.EPS2(J).AND.BOOGD.LT.EPSD2(J) )                    PBGN2740
     &            CVCTR(J) = CVCTR(J)*0.5                            PBGN2750
 4400 CONTINUE                                                       PBGN2760
      IF( ABS(BANK) .GT. DUCK1   .OR. ABS(BANKD) .GT. DUCK2 )        PBGN2770
     &     CVCTR( 2 ) = CVCTR( 2 ) * DUCK3                           PBGN2780
 4300 CALL TURKEY(0)                                                 PBGN2790
C    FLAG2 = .TRUE.  PRINT EVERYTHING                                PBGN2800
```

```
      KKK = KKK + 01                                                    PBGN2810
      IF( KKK .NE. NF9 ) GOTO 630                                       PBGN2820
      KKK = 00                                                          PBGN2830
      WRITE(6,45)                                                       PBGN2840
      WRITE(6,2)TIME                                                    PBGN2850
      DO 445   J = 1 , 15                                               PBGN2860
      WRITE(6,5) J,SV(J),SVD(J),TSV(J)                                  PBGN2870
  445 CONTINUE                                                          PBGN2880
      WRITE(6,42)(CVCTR(J),J=1,4)                                       PBGN2890
      DO 900  J = 1 , 4                                                 PBGN2900
      ZONK( NO , J ) = CVCTR( J )                                       PBGN2910
  900  CONTINUE                                                         PBGN2920
      IF( FLAG2 ) GOTO 630                                              PBGN2930
      WRITE(6,6)(ERVCTR(J),J=1,3)                                       PBGN2940
      PRINT52                                                           PBGN2950
      ASSIGN 446 TO KRASH                                               PBGN2960
      GOTO 411                                                          PBGN2970
  446  CONTINUE                                                         PBGN2980
      PRINT51,BANK,YANK,R,SIGMA,ERR,BANKD,YANKD,RDOT,SIGMAD,ERRD        PBGN2990
     &,BANKDD,YANKDD,RDDOT,SIGMDD,ERRDD                                 PBGN3000
      IF(  NO .GT. 150   ) GOTO  5000                                   PBGN3010
      NOPN = NO + 304                                                   PBGN3020
      NOPM = NO + 152                                                   PBGN3030
      BV( NOPN ) = BANK                                                 PBGN3040
      YV( NOPN ) = YANK                                                 PBGN3050
      RV( NOPN ) = RDIPH                                                PBGN3060
      TV( NOPN ) = SIGMA                                                PBGN3070
      BV( NOPM ) = BANKDD                                               PBGN3080
      YV( NOPM ) = YANKDD                                               PBGN3090
      RV( NOPM ) = RDDOT                                                PBGN3100
      TV( NOPM ) = SIGMAD                                               PBGN3110
 5000   CONTINUE                                                        PBGN3120
      PRINT9,XHUD,XHUDD,YHUD,YHUDD                                      PBGN3130
      GOTCHA = 0.0                                                      PBGN3140
        IF( ABS(ERR).LT.EPSAR .AND. ABS(ERRD).LT.EPSARD ) GOTCHA = 1.0  PBGN3150
      IF( GOTCHA .EQ. 1.0 ) PRINT 11                                    PBGN3160
  630 CONTINUE                                                          PBGN3170
```

```
      DO          4500        J=1,15
      ST(J) = SV(J)
      SVDS(J) = SVD(J)
 4500 SV(J) = SV(J) + SVD(J)*AICH
      TIME = TIME + AICH
      CALL TURKEY(0)
      DO          4600        J=1,15
 4600 SV(J) = ST(J) + AICH2*( SVDS(J)+SVD(J) )
      GOTO 400
 4700  CALL ESCAPE(0)
      CALL SSIGHT(0)
```

```
      RETURN
 1    FORMAT(10X,3I10)
 2    FORMAT('0',8X,'ELAPSED TIME',F10.3)
 3    FORMAT(A6,4X,2I5,6E10.2)
 4    FORMAT(A10,7E10.2/(10X,7E10.2))
 5    FORMAT( 9X,4X,I2,1P 2E15.4,10X,E15.4)
 6    FORMAT( 9X,'ERROR VECTOR = ',1P 3E15.4)
 7    FORMAT(1X,A6,4X,2I5,6E10.2)
 8    FORMAT(1X,A10,7E10.2/(10X,7E10.2))
 9    FORMAT(9X,6HPIPPER1P4E15.4)
 11   FORMAT(9X,6HGOTCHA)
 42   FORMAT( 9X,'CONTROL VECTOR =',1P4E15.4)
 45   FORMAT(9X 80(1H*))
 48   FORMAT(9X 10HRANGE =      ,F12.1)
 51     FORMAT(9X,1P5E15.4)
 52   FORMAT(9X,7X,5H BANK10X,5H YANK10X,5HRANGE10X,5HSIGMA10X,5HERROR)
      END
```

```
      SUBROUTINE TURKEY(NTRY)                                           TURK  10
      REAL    MASS,MACH                                                 TURK  20
      COMMON    /TDATA/    TDUM1(550),    TDUM2(4700)                   TURK  30
      COMMON /IDATA1/ IDUM1(131)                                        TURK  40
      COMMON/PIGLET/DUM(25)                                             TURK  50
      COMMON /FROGS/TWB(3,3),TWE(3,3),TWBD(3,3),W(3,3),WD(3,3)          TURK  60
      COMMON /STATE/ SSV(80)                                            TURK  70
      DIMENSION X(15),XD(15),XXD(3),XXDD(3)                             TURK  80
     &               ,SV(30),SVD(15),CVCTR(4)                           TURK  90
     &               ,XDD(3),TBE(3,3)                                   TURK 100
     &          ,QRK(3)                                                 TURK 110
      DIMENSION ALR7M(8),ALR7H(6),ALR7A(5),CLRT(240)                    TURK 120
      DIMENSION  ACYBM(8),  ACYBH(6),  ACYBA(5),  CYBT(240)             TURK 130
      DIMENSION  ACLBM(8),  ACLBH(6),  ACLBA(5),  CLBT(240)             TURK 140
      DIMENSION  ALDRM(3),  ALDRH(6),  ALDRA(5),  CLDRT(240)            TURK 150
      DIMENSION  AMDSM(8),  AMDSH(6),  AMDSA(5),  CMDST(240)            TURK 160
      DIMENSION  ANB7M(8),  ANB7H(6),  ANB7A(5),  CNBT(240)             TURK 170
      DIMENSION  AYP7M(8),  AYP7H(6),  AYP7A(5),  CYPT(240)             TURK 180
      DIMENSION  ANP7M(8),  ANP7H(6),  ANP7A(5),  CNPT(240)             TURK 190
      DIMENSION  AYDRM(8),  AYDRH(6),             CYDRT(48)             TURK 200
      DIMENSION  AYR7M(8),  AYR7H(6),  AYR7A(5),  CYRT(240)             TURK 210
      DIMENSION  AZQ7M(8),  AZQ7H(6),             CZQT(48)              TURK 220
      DIMENSION  AZADM(8),  AZADH(6),             CZADT(48)             TURK 230
      DIMENSION  AZDSM(8),  AZDSH(6),  AZDSA(5),  CZDST(240)            TURK 240
      DIMENSION  ALDAM(8),  ALDAH(6),  ALDAA(5),  CLDAT(240)            TURK 250
      DIMENSION  AMADM(8),  AMADH(6),             CMADT(48)             TURK 260
      DIMENSION  AMQ7M(8),  AMQ7H(6),             CMQT(48)              TURK 270
      DIMENSION  ANDRM(8),  ANDRH(6),             CNDRT(48)             TURK 280
      DIMENSION  ALP7M(8),  ALP7H(6),  ALP7A(5),  CLPT(240)             TURK 290
      DIMENSION  ANDAM(8),  ANDAH(6),  ANDAA(5),  CNDAT(240)            TURK 300
      DIMENSION  AYDAM(8),                        CYDAT(8)              TURK 310
      DIMENSION  ANR7M(8),  ANR7H(6),  ANR7A(5),  CNRT(240)             TURK 320
      DIMENSION  ACL7M(8),  ACL7H(6),  ACL7A(5),  CLT(240)              TURK 330
      DIMENSION  ACM7M(8),  ACM7H(6),  ACM7A(5),  CMT(240)              TURK 340
      DIMENSION  ADT7M(8),  ADT7R(6)                                    TURK 350
      DIMENSION  CRHOT(70),   CDT(80)                                   TURK 360
      DIMENSION  ALTRM(8),  ALTRH(6),  ALTRA(5),  CLTRT(240)            TURK 370
      DIMENSION  ARHO(100)                                              TURK 380
      EQUIVALENCE (DUM(1),DE),(DUM(2),S),(DUM(3),B),(DUM(4),MASS),      TURK 390
     1 (DUM(5),G),(DUM(6),RC),(DUM(7),TIXX),(DUM(8),TIYY),(DUM(9),TIZZ),TURK 400
```

BEST AVAILABLE COPY

```
      2 (DUM(10),TIXZ),(DUM(11),ALW),(DUM(12),AMT),                    TURK 410
      3 (DUM(13),DRUL),(DUM(14),DRLL),(DUM(15),DDSUL),(DUM(16),DDSLL), TURK 420
      4 (DUM(17),DAUL),(DUM(18),DALL),(DUM(19),THRST1),                TURK 430
      5 (DUM(20),THRST2),(DUM(21),THRST3),(DUM(22),THRST4)             TURK 440
      &            ,(DUM(23),ALPMX),(DUM(24),BETMX) ,(DUM(25),GMX)      TURK 450
       EQUIVALENCE(X(7),V),(X(8),PHI),(X(9),THETA),(X(10),PSI)         TURK 460
      &            ,(X(11),ALPHA),(X(12),BETA),(X(13),P),(X(14),Q)      TURK 470
      &            ,(X(15),R)                                           TURK 480
      &            ,(XD(1),XXD(1)),(XD(4),XXDD(1)),(XD(7),VD)           TURK 490
      &            ,(XD(8),PHID),(XD(9),THETAD),(XD(10),PSID),          TURK 500
      &            (XD(11),ALPHAD),(XD(12),BETAD),(XD(13),PD)           TURK 510
      &            ,(XD(14),QD),(XD(15),RD)                             TURK 520
      &            ,(GSV(17),TBE(1,1)),(GSV(26),RHO)                    TURK 530
      &            ,(SV(1),X(1)),(SVD(1),XD(1)),(SV(26),CVCTR(1))       TURK 540
       EQUIVALENCE(ACYBM(1),TDUM1(   1)),(ACYBH(1),TDUM1(   9))        TURK 550
       EQUIVALENCE(ACYBA(1),TDUM1(  15)),(ACLBM(1),TDUM1(  20))        TURK 560
       EQUIVALENCE(ACLBH(1),TDUM1(  28)),(ACLBA(1),TDUM1(  34))        TURK 570
       EQUIVALENCE(ALDRM(1),TDUM1(  39)),(ALDRH(1),TDUM1(  47))        TURK 580
      &       ,(SVD(1),GSV(62))                                        TURK 590
      &     ,(GSV(2),SV(1))                                            TURK 600
       EQUIVALENCE(ALDRA(1),TDUM1(  53)),(ALR7M(1),TDUM1(  58))        TURK 610
       EQUIVALENCE(ALR7H(1),TDUM1(  66)),(ALR7A(1),TDUM1(  72))        TURK 620
       EQUIVALENCE(AMDSM(1),TDUM1(  77)),(AMDSH(1),TDUM1(  85))        TURK 630
       EQUIVALENCE(AMDSA(1),TDUM1(  91)),(ANB7M(1),TDUM1(  96))        TURK 640
       EQUIVALENCE(ANB7H(1),TDUM1( 104)),(ANB7A(1),TDUM1( 110))        TURK 650
       EQUIVALENCE(AYP7M(1),TDUM1( 115)),(AYP7A(1),TDUM1( 123))        TURK 660
       EQUIVALENCE(AYP7H(1),TDUM1( 128)),(ANP7M(1),TDUM1( 134))        TURK 670
       EQUIVALENCE(ANP7A(1),TDUM1( 142)),(ANP7H(1),TDUM1( 147))        TURK 680
       EQUIVALENCE(AYDRM(1),TDUM1( 153)),(AYDRH(1),TDUM1( 161))        TURK 690
       EQUIVALENCE(AYR7M(1),TDUM1( 167)),(AYR7H(1),TDUM1( 175))        TURK 700
       EQUIVALENCE(AZQ7M(1),TDUM1( 181)),(AZQ7H(1),TDUM1( 189))        TURK 710
       EQUIVALENCE(AZADM(1),TDUM1( 195)),(AZADH(1),TDUM1( 203))        TURK 720
       EQUIVALENCE(AZDSM(1),TDUM1( 209)),(AZDSA(1),TDUM1( 217))        TURK 730
       EQUIVALENCE(AZDSH(1),TDUM1( 222)),(ALDAM(1),TDUM1( 228))        TURK 740
       EQUIVALENCE(ALDAH(1),TDUM1( 236)),(AMADM(1),TDUM1( 242))        TURK 750
       EQUIVALENCE(AMADH(1),TDUM1( 250)),(AMQ7M(1),TDUM1( 256))        TURK 760
       EQUIVALENCE(AMQ7H(1),TDUM1( 264)),(ANDRM(1),TDUM1( 270))        TURK 770
       EQUIVALENCE(ANDRH(1),TDUM1( 278))                               TURK 780
       EQUIVALENCE(ALP7H(1),TDUM1( 291)),(ANDAM(1),TDUM1( 297))        TURK 790
       EQUIVALENCE(ANDAA(1),TDUM1( 305)),(ANDAH(1),TDUM1( 310))        TURK 800
```

```
      EQUIVALENCE(AYDAM(1),TDUM1( 316)),(ANR7M(1),TDUM1( 324))          TURK 810
      EQUIVALENCE(ACL7A(1),TDUM1( 332)),(ACL7M(1),TDUM1( 337))          TURK 820
      EQUIVALENCE(ACL7H(1),TDUM1( 345)),(ACM7A(1),TDUM1( 351))          TURK 830
      EQUIVALENCE(ACM7M(1),TDUM1( 356)),(ACM7H(1),TDUM1( 364))          TURK 840
      EQUIVALENCE(ADT7R(1),TDUM1(370)),(ADT7M(1),TDUM1(380))            TURK 850
      EQUIVALENCE(ALTRA(1),TDUM1( 388)),(ALTRM(1),TDUM1( 393))          TURK 860
      EQUIVALENCE( ALTRH(1),TDUM1(401)),(ARHO(1),TDUM1(407))            TURK 870
      EQUIVALENCE(AYR7A(1),TDUM1( 515)),(ALDAA(1),TDUM1( 520))          TURK 880
      EQUIVALENCE(ALP7A(1),TDUM1( 525)),(ANP7H(1),TDUM1( 530))          TURK 890
      EQUIVALENCE(ANR7A(1),TDUM1( 536))                                 TURK 900
      EQUIVALENCE(ALP7M(1),TDUM1( 541))                                 TURK 910
      EQUIVALENCE( CYBT(1)  ,TDUM2(   1)),  ( CLBT(1)  ,TDUM2( 241))    TURK 920
      EQUIVALENCE( CLDRT(1) ,TDUM2( 481)),  ( CLRT(1)  ,TDUM2( 721))    TURK 930
      EQUIVALENCE( CMDST(1) ,TDUM2( 961)),  ( CNBT(1)  ,TDUM2(1201))    TURK 940
      EQUIVALENCE( CYPT(1)  ,TDUM2(1441)),  ( CNPT(1)  ,TDUM2(1681))    TURK 950
      EQUIVALENCE( CYDRT(1) ,TDUM2(1921))                               TURK 960
      EQUIVALENCE( CZQT(1)  ,TDUM2(2017)),  ( CZADT(1),TDUM2(2065))     TURK 970
      EQUIVALENCE( CZDST(1) ,TDUM2(2113))                               TURK 980
      EQUIVALENCE( CMADT(1) ,TDUM2(2401)),  ( CMQT(1)  ,TDUM2(2449))    TURK 990
      EQUIVALENCE( CNDRT(1) ,TDUM2(2497))                               TURK1000
      EQUIVALENCE( CNDAT(1) ,TDUM2(2593)),  ( CYDAT(1),TDUM2(2833))     TURK1010
      EQUIVALENCE( CLT(1)   ,TDUM2(2849))                               TURK1020
      EQUIVALENCE( CMT(1)           ,  TDUM2(3089) )                    TURK1030
      EQUIVALENCE( CDT(1)  ,TDUM2(3329)),(CRHOT(1),TDUM2(3409))         TURK1040
      EQUIVALENCE( CLTRT(1)         ,  TDUM2(3479) )                    TURK1050
      EQUIVALENCE( CYRT(1)  ,TDUM2(3727)),  (CLDAT(1),TDUM2(3967))      TURK1060
      EQUIVALENCE( CLPT(1)  ,TDUM2(4207)),  (CNRT(1) ,TDUM2(4447))      TURK1070
      EQUIVALENCE(NCYBM,IDUM1(  1)),(NCYBH,IDUM1(  2)),(NCYBA,IDUM1( 3))TURK1080
      EQUIVALENCE(NCLBM,IDUM1(  4)),(NCLBH,IDUM1(  5)),(NCLBA,IDUM1( 6))TURK1090
      EQUIVALENCE(NLDRM,IDUM1(  7)),(NLDRH,IDUM1(  8)),(NLDRA,IDUM1( 9))TURK1100
      EQUIVALENCE(NLR7M,IDUM1( 10)),(NLR7H,IDUM1( 11)),(NLR7A,IDUM1(12))TURK1110
      EQUIVALENCE(NMDSM,IDUM1( 13)),(NMDSH,IDUM1( 14)),(NMDSA,IDUM1(15))TURK1120
      EQUIVALENCE(NNB7M,IDUM1( 16)),(NNB7H,IDUM1( 17)),(NNB7A,IDUM1(18))TURK1130
      EQUIVALENCE(NYP7M,IDUM1( 19)),(NYP7A,IDUM1( 20)),(NYP7H,IDUM1(21))TURK1140
      EQUIVALENCE(NNP7M,IDUM1( 22)),(NNP7A,IDUM1( 23)),(NNP7H,IDUM1(24))TURK1150
      EQUIVALENCE(NYDRM,IDUM1( 25)),(NYDRH,IDUM1( 26))                  TURK1160
      EQUIVALENCE(NYR7M,IDUM1( 27)),(NYR7H,IDUM1( 28))                  TURK1170
      EQUIVALENCE(NZQ7M,IDUM1( 29)),(NZQ7H,IDUM1( 30))                  TURK1180
      EQUIVALENCE(NZADM,IDUM1( 31)),(NZADH,IDUM1( 32))                  TURK1190
      EQUIVALENCE(NZDSM,IDUM1( 33)),(NZDSA,IDUM1( 34)),(NZDSH,IDUM1(35))TURK1200
```

```
      EQUIVALENCE(NLDAM,IDUM1( 36)),(NLDAH,IDUM1( 37))                  TURK1210
      EQUIVALENCE(NMADM,IDUM1( 38)),(NMADH,IDUM1( 39))                  TURK1220
      EQUIVALENCE(NMO7M,IDUM1( 40)),(NMO7H,IDUM1( 41))                  TURK1230
      EQUIVALENCE(NNDRM,IDUM1( 42)),(NNDRH,IDUM1( 43))                  TURK1240
      EQUIVALENCE(NLP7M,IDUM1( 44)),(NLP7H,IDUM1( 45))                  TURK1250
      EQUIVALENCE(NNDAM,IDUM1( 46)),(NNDAA,IDUM1( 47)),(NNDAH,IDUM1(67))TURK1260
      EQUIVALENCE(NYDAM,IDUM1( 48))                                     TURK1270
      EQUIVALENCE(NNR7M,IDUM1( 49))                                     TURK1280
      EQUIVALENCE(NCL7A,IDUM1( 50)),(NCL7M,IDUM1( 51)),(NCL7H,IDUM1(52))TURK1290
      EQUIVALENCE(NCM7A,IDUM1( 53)),(NCM7M,IDUM1( 54)),(NCM7H,IDUM1(55))TURK1300
      EQUIVALENCE(NYR7A,IDUM1( 56)),(NLDAA,IDUM1( 57)),(NLP7A,IDUM1(58))TURK1310
      EQUIVALENCE(NDT7R,IDUM1(59)),(NDT7M,IDUM1(60))                    TURK1320
      EQUIVALENCE(NRHO ,IDUM1( 61))                                     TURK1330
      EQUIVALENCE(NNR7H,IDUM1( 62)),(NNR7A,IDUM1( 63))                  TURK1340
      EQUIVALENCE  ( INDTBE         ,  IDUM1( 73))                      TURK1350
      EQUIVALENCE  ( IRETAF         ,IDUM1( 83))                        TURK1360
      EQUIVALENCE  ( INDERR    ,  IDUM1( 98))                           TURK1370
      EQUIVALENCE(NLTRH ,IDUM1(101)),(NLTRM ,IDUM1(102))                TURK1380
      EQUIVALENCE(NLTRA ,IDUM1(103))                                    TURK1390
      DATA RCP / 968.12 /                                               TURK1400
      DATA DDR /57.29578/                                               TURK1410
      DATA JUMP/0/                                                      TURK1420
      IF( NTRY.GE.00 ) GOTO 2000                                        TURK1430
```

```
      IF( JUMP.GT.00 ) GOTO 1000                                    TKIN  10
      JUMP = 01                                                     TKIN  20
      READ1,IDUM1                                                   TKIN  30
      READ2,TDUM1                                                   TKIN  40
      READ2,TDUM2                                                   TKIN  50
      READ2,DUM                                                     TKIN  60
      PRINT 3, DUM                                                  TKIN  70
      DO  12  I = 13 , 18                                           TKIN  80
      DUM( I ) = DUM( I ) / DDR                                     TKIN  90
   12 CONTINUE                                                      TKIN 100
      DUM( 01 ) = DUM( 01 ) / DDR                                   TKIN 110
      DUM( 11 ) = DUM( 11 ) / DDR                                   TKIN 120
      DUM( 23 ) = DUM( 23 ) / DDR                                   TKIN 130
      DUM( 24 ) = DUM( 24 ) / DDR                                   TKIN 140
      COSDE  = COS( DE )                                            TKIN 150
      SINDE  = SIN( DE )                                            TKIN 160
      ACCMX = G*GMX                                                 TKIN 170
 1000 IF( NTRY .LT. -01 ) GOTO 1100                                 TKIN 180
      READ2,MASS,TIXX,TIYY,TIZZ,TIXZ                                TKIN 190
      PRINT 3,MASS,TIXX,TIYY,TIZZ,TIXZ                              TKIN 200
      GM =   MASS * G                                               TKIN 210
      THMEAN = 0.25*( THRST1+THRST2+THRST3+THRST4 )                 TKIN 220
      CVCTR(1) = DAUL                                               TKIN 230
      CVCTR(2) = DDSUL                                              TKIN 240
      CVCTR(3) = THRST1 - THMEAN                                    TKIN 250
      CVCTR(4) = DRUL                                               TKIN 260
      DO       1050      J=1,3                                      TKIN 270
      WD(J,J) = 0.0                                                 TKIN 280
 1050 W(J,J) = 0.0                                                  TKIN 290
      XXI = TIYY - TIZZ                                             TKIN 300
      YYI = TIZZ - TIXX                                             TKIN 310
      ZZI = TIXX - TIYY                                             TKIN 320
      DET = TIXX*TIZZ - TIXZ**2                                     TKIN 330
      RETURN                                                        TKIN 340
 1100 CVCTR(1) = DALL                                               TKIN 350
      CVCTR(2) = DDSLL                                              TKIN 360
      CVCTR(3) = THRST4 - THMEAN                                    TKIN 370
      CVCTR(4) = DRLL                                               TKIN 380
      RETURN                                                        TKIN 390
 2000 IF( NTRY .EQ. 02 ) GOTO 3000                                  TKIN 400
```

```
      H = - X( 3 )
      W(1,2) = -R
      W(1,3)   = +Q
      W(2,3) = -P
      W(2,1) =   - W(1,2)
      W(3,1) = - W(1,3)
      W(3,2) = - W(2,3)
      COSTHE = COS( THETA )
      SINTHE = SIN( THETA )
      COSPHI = COS( PHI )
      SINPHI = SIN( PHI )
      COSPSI = COS( PSI )
      SINPSI = SIN( PSI )
         TBE(1,1) = COSPSI * COSTHE
         TBE(2,1) = SINPSI * COSTHE
         TBE(3,1) =   - SINTHE
         TBE(1,2) = COSPSI * SINTHE * SINPHI - SINPSI * COSPHI
         TBE(2,2) = SINPSI * SINTHE * SINPHI + COSPSI * COSPHI
         TBE(3,2) = COSTHE * SINPHI
         TBE(1,3) = COSPSI * SINTHE * COSPHI + SINPSI * SINPHI
         TBE(2,3) = SINPSI * SINTHE * COSPHI - COSPSI * SINPHI
         TBE(3,3) = COSTHE * COSPHI
      SUM = 0.0
      COSALP = COS( ALPHA )
      SINALP = SIN( ALPHA )
      COSBTA = COS( BETA )
      SINBTA = SIN( BETA )
         TWB(1,1) =  COSALP * COSBTA
         TWB(1,2) = - COSALP * SINBTA
         TWB(1,3) =   SINALP
         TWB(2,1) = SINBTA
         TWB(2,2) = COSBTA
         TWB(2,3) = 0.0
         TWB(3,1) = SINALP * COSBTA
         TWB(3,2) =   - SINALP * SINBTA
         TWB(3,3) =   - COSALP
      DO  100 I = 1 , 3
      DO  100 J = 1 , 3
      SUM = 0.0
      DO 101  K = 1 , 3
```

```
      SUM = SUM + TBE(I,K) * TWB(K,J)
101   CONTINUE
        TWE(I,J) = SUM
100   CONTINUE
      IF( NTRY.EQ.00 ) GOTO 2100
      IF( H .LT. 36000.0 ) MACH = V/(49.0 * SQRT(518.7 - H * 3.565E-3))
      IF( H .GT. 36000.0 ) MACH = V / BCP
      ALPHAW = ( ALPHA + ALW ) * DDR
C   ALPHAW IS NOW IN DEGREES .......................
      CALL LOOK3
     $(CLDR,H,MACH,ALPHAW,ALDRH,NLDRH,ALDRM,NLDRM,ALDRA,NLDRA,CLDRT,0)
      CALL LOOK3
     $(CLR,H,MACH,ALPHAW,ALR7H,NLR7H,ALR7M,NLR7M,ALR7A,NLR7A,CLRT,0)
      CALL LOOK3
     $(CMDS,H,MACH,ALPHAW,AMDSH,NMDSH,AMDSM,NMDSM,AMDSA,NMDSA,CMDST,0)
      CALL LOOK3
     $(CYP,H,MACH,ALPHAW,AYP7H,NYP7H,AYP7M,NYP7M,AYP7A,NYP7A,CYPT,0)
      CALL LOOK3
     $(CNP,H,MACH,ALPHAW,ANP7H,NNP7H,ANP7M,NNP7M,ANP7A,NNP7A,CNPT,0)
      CALL LOOK2
     $(CYDR,H,MACH,AYDRH,NYDRH,AYDRM,NYDRM,CYDRT,0)
      CALL LOOK3
     $(CYR,H,MACH,ALPHAW,AYR7H,NYR7H,AYR7M,NYR7M,AYR7A,NYR7A,CYRT,0)
      CALL LOOK2
     $(CZQ,H,MACH,AZQ7H,NZQ7H,AZQ7M,NZQ7M,CZQT,0)
      CALL LOOK2
     $(CZAD,H,MACH,AZADH,NZADH,AZADM,NZADM,CZADT,0)
      CALL LOOK3
     $(CZDS,H,MACH,ALPHAW,AZDSH,NZDSH,AZDSM,NZDSM,AZDSA,NZDSA,CZDST,0)
      CALL LOOK3
     $(CLDA,H,MACH,ALPHAW,ALDAH,NLDAH,ALDAM,NLDAM,ALDAA,NLDAA,CLDAT,0)
      CALL LOOK2
     $(CMAD,H,MACH,AMADH,NMADH,AMADM,NMADM,CMADT,0)
      CALL LOOK2
     $(CMQ,H,MACH,AMQ7H,NMQ7H,AMQ7M,NMQ7M,CMQT,0)
      CALL LOOK2
     $(CNDR,H,MACH,ANDRH,NNDRH,ANDRM,NNDRM,CNDRT,0)
      CALL LOOK3
     $(CLP,H,MACH,ALPHAW,ALP7H,NLP7H,ALP7M,NLP7M,ALP7A,NLP7A,CLPT,0)
      CALL LOOK3
```

BEST AVAILABLE COPY

```
       $(CNDA,H,MACH,ALPHAW,ANDAH,NNDAH,ANDAM,NNDAM,ANDAA,NNDAA,CNDAT,0)   TKIN1210
        CALL LOOK1(CYDA,MACH,AYDAM,NYDAM,CYDAT,0)                          TKIN1220
        CALL LOOK3                                                         TKIN1230
       $(CNR,H,MACH,ALPHAW,ANR7H,NNR7H,ANR7M,NNR7M,ANR7A,NNR7A,CNRT,0)     TKIN1240
        CALL LOOK3                                                         TKIN1250
       $(CL,H,MACH,ALPHAW,ACL7H,NCL7H,ACL7M,NCL7M,ACL7A,NCL7A,CLT,0)       TKIN1260
        CALL LOOK3                                                         TKIN1270
       $(CM,H,MACH,ALPHAW,ACM7H,NCM7H,ACM7M,NCM7M,ACM7A,NCM7A,CMT,0)       TKIN1280
        CALL LOOK1( RHO,H,ARHO,NRHO,CRHOT,0)                               TKIN1290
        QB = 0.5 * RHO * V * V                                             TKIN1300
        CALL LOOK3                                                         TKIN1310
       $(CYB,H,MACH,ALPHAW,ACYBH,NCYBH,ACYBM,NCYBM,ACYBA,NCYBA,CYBT,0)     TKIN1320
        CALL LOOK3                                                         TKIN1330
       $(CLB,H,MACH,ALPHAW,ACLBH,NCLBH,ACLBM,NCLBM,ACLBA,NCLBA,CLBT,0)     TKIN1340
        CALL LOOK3                                                         TKIN1350
       $(CNB,H,MACH,ALPHAW,ANB7H,NNB7H,ANB7M,NNB7M,ANB7A,NNB7A,CNBT,0)     TKIN1360
        CALL LOOK3                                                         TKIN1370
       $(CLTR,H,MACH,ALPHAW,ALTRH,NLTRH,ALTRM,NLTRM,ALTRA,NLTRA,CLTRT,0)   TKIN1380
        ALPHAW=ALPHAW/DDR                                                  TKIN1390
C   ALPHAW IS NOW IN RADIANS .......................                       TKIN1400
        CALL LOOK2(CD,MACH,ALPHAW,ADT7M,NDT7M,ADT7R,NDT7R,CDT,0)           TKIN1410
 2100     CONTINUE                                                         TKIN1420
        ALPSE = ALPHA + DE                                                 TKIN1430
        CALPSE = COS(ALPSE)                                                TKIN1440
        SALPSE = SIN(ALPSE)                                                TKIN1450
        SINGAM = - TWE(3,1)                                                TKIN1460
        TWOV = 2.00*V                                                      TKIN1470
        BD2V = B / TWOV                                                    TKIN1480
        QUAN1 =  P * COSALP + R * SINALP                                   TKIN1490
        QUAN2 =  R * COSALP - P * SINALP                                   TKIN1500
        QBS = QB*S                                                         TKIN1510
        QBSRC = QBS*RC                                                     TKIN1520
        DRAG = QBS*CD                                                      TKIN1530
        RCD2V = RC / TWOV                                                  TKIN1540
        STUFF = RCD2V * QBS                                                TKIN1550
        VM = V * MASS                                                      TKIN1560
        B1 =  + QBS   *  TWB( 3,1 )                                        TKIN1570
        B2 =  + QBS   *  TWB( 1,1 )                                        TKIN1580
        B3 =  - QBS   *  TWB( 3,3 )                                        TKIN1590
        B4 =  - QBS   *  TWB( 1,3 )                                        TKIN1600
```

```
      T1 = ( CLP*QUAN1 + CLR*QUAN2 )*BD2V + CLB*BETA                    TKIN1610
      T2 = ( CNP*QUAN1 + CNR*QUAN2 )*BD2V + CNB*BETA                    TKIN1620
      STUFF1 = XXI*Q*R + TIXZ*Q*P + ( B2*T1 + B4*T2 )*B                 TKIN1630
      STUFF2 = ZZI*P*Q - TIXZ*R*Q +( B1*T1 + B2*T2 )*B                  TKIN1640
      A1 = ( STUFF1*TIZZ + STUFF2*TIXZ )/DET                            TKIN1650
      A2 = ( STUFF2*TIXX + STUFF1*TIXZ )/DET                            TKIN1660
      B1 = B1 / DET                                                     TKIN1670
      B2 = B2 / DET                                                     TKIN1680
      B3 = B3 / DET                                                     TKIN1690
      B4 = B4 / DET                                                     TKIN1700
      AJUNK = B1*TIXZ + B2*TIZZ                                         TKIN1710
      BJUNK = B3*TIXZ + B4*TIZZ                                         TKIN1720
      A11 = AJUNK*CLDR + BJUNK*CNDR                                     TKIN1730
      A12 = AJUNK*CLDA + BJUNK*CNDA                                     TKIN1740
      AJUNK = B2*TIXZ + B1*TIXX                                         TKIN1750
      BJUNK = B4*TIXZ + B3*TIXX                                         TKIN1760
      A21 = AJUNK*CLDR + BJUNK*CNDR                                     TKIN1770
      A22 = AJUNK*CLDA + BJUNK*CNDA                                     TKIN1780
      IF( NTRY.EQ.01 ) RETURN                                           TKIN1790
 3000 CONTINUE                                                          TKIN1800
      DA   = CVCTR( 1 )                                                 TKIN1810
      DDS = CVCTR(2)                                                    TKIN1820
      CDHM = CVCTR(3) + THMEAN                                          TKIN1830
      DR   = CVCTR( 4 )                                                 TKIN1840
      COM = CYDR*DR + CYB*BETA + CYDA*DA + BD2V*(CYP*QUAN1 + CYR*QUAN2) TKIN1850
      SFC = QBS*COM                                                     TKIN1860
      XXLF = QBS*(-CL + CZDS*DDS + RCD2V * CZQ * Q )                    TKIN1870
      VD = (COSBTA*(CDHM*CALPSE - DRAG) + SINBTA*SFC)/MASS - G*SINGAM   TKIN1880
      ALPHAD = (-CDHM*SALPSE - GM*TWE(3,3) + XXLF                       TKIN1890
     &       + VM * ( Q * COSBTA - SINBTA * QUAN1 ))                    TKIN1900
     2       / (VM * COSBTA - STUFF * CZAD )                            TKIN1910
      BETAD = (SINBTA*(-CDHM*CALPSE + DRAG) + COSBTA*SFC)/ VM           TKIN1920
     &       +   TWE(3,2) * G / V - QUAN2                               TKIN1930
      IF( ALPHA.GE.ALPMX .AND. ALPHAD.GE.0.0 ) ALPHAD = 0.0             TKIN1940
      IF( ALPHA.LE.-ALPMX .AND. ALPHAD.LE.0.0 ) ALPHAD = 0.0            TKIN1950
      IF( BETA.GE.BETMX .AND. BETAD.GE.0.0 ) BETAD = 0.0                TKIN1960
      IF( BETA.LE.-BETMX .AND. BETAD.LE.0.0 ) BETAD = 0.0               TKIN1970
      PD = A1 +(A11*DR + A12*DA)*B                                      TKIN1980
      RD = A2 +(A21*DR + A22*DA)*B                                      TKIN1990
      QJUNK = YYI*R*P + ( R*R - P*P )*TIXZ                              TKIN2000
```

```
     &          + QBSRC*( CM + RCD2V*( CMAD*ALPHAD + CMQ*G ) )              TKIN2010
      QD = ( QJUNK + QBSRC*CMDS*DDS + AMT*CDHM ) / TIYY                    TKIN2020
      PSID = ( R*COSPHI + Q*SINPHI ) / COSTHE                              TKIN2030
      THETAD = Q*COSPHI - R*SINPHI                                         TKIN2040
      PHID = P + PSID*SINTHE                                               TKIN2050
      QRK(1) = VD                                                          TKIN2060
      QRK(2) = V*( BETAD + QUAN2 )                                         TKIN2070
      QRK(3) =-V*COSBTA*( ALPHAD+QUAN1*SINBTA/COSBTA-Q )                   TKIN2080
      DO  401  I = 1 , 3                                                   TKIN2090
       SVD( I ) = SV( I+03 )                                               TKIN2100
      SUM = 0.0                                                            TKIN2110
      DO  402  J = 1 , 3                                                   TKIN2120
402    SUM = SUM + TWE(I,J) * QRK(J)                                       TKIN2130
401    SVD( I + 3 ) = SUM                                                  TKIN2140
      WD(1,2) = -RD                                                        TKIN2150
      WD(1,3) = +QD                                                        TKIN2160
      WD(2,3) = -PD                                                        TKIN2170
      WD(2,1) = - WD(1,2)                                                  TKIN2180
      WD(3,1) = - WD(1,3)                                                  TKIN2190
      WD(3,2) = -WD(2,3)                                                   TKIN2200
      SUM = SQRT( XXDD(1)**2+XXDD(2)**2+XXDD(3)**2 )                       TKIN2210
      IF( SUM.LE.ACCMX) GOTO 4003                                          TKIN2220
      SUM = ACCMX/SUM                                                      TKIN2230
      DO        4304       J = 1,3                                         TKIN2240
4304  XXDD(J) = XXDD(J)*SUM                                                TKIN2250
4003   CONTINUE                                                            TKIN2260
      RETURN                                                               TKIN2270
1     FORMAT(20I4)                                                         TKIN2280
2     FORMAT(8E10.2)                                                       TKIN2290
3     FORMAT(' TURKEY',1P10E12.3)                                          TKIN2300
      END                                                                  TKIN2310
```

```
      SUBROUTINE SSIGHT( NTRY )                                         SIGH  10
C  VREL = RELATIVE VELOCITY   (INERTIAL SYSTEM)                         SIGH  20
C  PREL = RELATIVE POSITION                                             SIGH  30
C  AREA = RELATIVE ACCELERATION                                         SIGH  40
C  RELPT = TOTAL RELATIVE POSITION                                      SIGH  50
C  RANGE = RANGE (HFH-HIH)                                              SIGH  60
C  UBAR = AVE PROJECTILE VELOCITY                                       SIGH  70
C  TF = TIME OF FLIGHT                                                  SIGH  80
C  RPI = ITERATIVE RANGE VECTOR TO TARGET                               SIGH  90
C  RPIT = TOTAL RPI VECTOR                                              SIGH 100
C  AVELA = VELOCITY VECTOR OF ATTACKER IN ATTACKER BODY COORDINATES     SIGH 110
C  AACCA = ATTACKER ACCELERATION VECTOR IN ATTACKER BODY COORDINATES    SIGH 120
C  AVPRJ = PROJECTILE VELOCITY VECTOR IN ATTACKER BODY COORDINATES      SIGH 130
C  PFP = PROJECTILE FUTURE POSITION (IMPACT POINT)  IN ATTACKER BODY COOSIGH 140
      COMMON/GUN/VELMUZ,GUNALP,GHR,DA,DE                                SIGH 150
      DIMENSION VREL(3),PREL(3),AREL(3),RPI(3),UNITZ(3)                 SIGH 160
      DIMENSION POSA(3),VELA(3),ACCA(3),POST(3),VELT(3),ACCT(3)         SIGH 170
      DIMENSION ATARG(3),PFP(3),VTARG(3),PTARG(3)                       SIGH 180
      DIMENSION AVPRJ(3),AACCA(3),AVELA(3)                              SIGH 190
     &          ,EULER(3,3)                                             SIGH 200
      COMMON /PILOTS/ BOOGIE(56),PREL,VREL                              SIGH 210
      COMMON /PIGLET/ BROOD(25)                                         SIGH 220
      COMMON /STATE/GSV(80)                                             SIGH 230
      EQUIVALENCE(GSV(2),POSA(1)),(GSV(5),VELA(1)),(GSV(65),ACCA(1))    SIGH 240
     &                ,(BROOD(5),GEE)                                   SIGH 250
     &                ,(GSV(14),P),(GSV(15),Q),(GSV(16),R)              SIGH 260
     &                ,(GSV(26),RHO),(GSV(1),TIME),(GSV(17),EULER(1,1)) SIGH 270
     &                ,(GSV(32),POST(1)),(GSV(35),VELT(1)),(GSV(8),VELAT) SIGH 280
     &                ,(GSV(39),ACCT(1)),(GSV(31),EGNAR)                SIGH 290
     &               ,(GSV(77),HUDX),(GSV(78),HUDY),(GSV(79),HUDXD)     SIGH 300
     &                ,(GSV(80),HUDYD)                                  SIGH 310
      COMMON /SIGHTS/ HALP,SIG                                          SIGH 320
      DATA UNITZ/0.,0.,1./                                              SIGH 330
      DATA MXLP/20/                                                     SIGH 340
      IF( NTRY.GE.00 )GOTO 5005                                         SIGH 350
      TOLD = TIME                                                       SIGH 360
      READ  1,LABEL,IORD,VELMUZ,GUNALP,GHR,DA,DE,HALP,SIG               SIGH 370
      PRINT 2,LABEL,IORD,VELMUZ,GUNALP,GHR,DA,DE,HALP,SIG               SIGH 380
      HALP = HALP/57.3                                                  SIGH 390
      GUNALP = GUNALP/57.3                                              SIGH 400
```

```
      IF( IORD.NE.03 ) GOTO 5701                                        SIGH 410
 5005 IF( IORD.EQ.03 ) GOTO 5500                                        SIGH 420
      VPROJ = VELMUZ + VELAT                                            SIGH 430
      CALL PROJ(ACOEF1,CDH1,CDH2,POSA(3) )                              SIGH 440
      RANGE = EGNAR                                                     SIGH 450
      DO   30   LOOP = 1,MXLP                                           SIGH 460
      CON1 = CDH1*RANGE                                                 SIGH 470
      CON2 = CDH1 * VPROJ + CDH2                                        SIGH 480
      CON3 = ACOEF1 * CDH1                                              SIGH 490
      CON4 = 1.0-EXP(-ACOEF1*CDH1*RANGE)                                SIGH 500
      UBAR = (1.0/CON1)*(CON2/CON3*CON4-CDH2*RANGE)                     SIGH 510
      IF( UBAR .GT. 0. ) GO TO 7003                                     SIGH 520
      PRINT 7004, RANGE                                                 SIGH 530
      GO TO 9999                                                        SIGH 540
 7003 CONTINUE                                                          SIGH 550
      TF = RANGE/UBAR                                                   SIGH 560
 7002 FORMAT(11F11.4)                                                   SIGH 570
      DO   40   I = 1,3                                                 SIGH 580
      RPI(I) = PREL(I) + VELT(I)*TF                                     SIGH 590
      IF( IORD .EQ. 1 ) GO TO 200                                       SIGH 600
      RPI(I) = RPI(I) + ACCT(I) * .5 * TF * TF                          SIGH 610
  200 CONTINUE                                                          SIGH 620
   40 CONTINUE                                                          SIGH 630
      RPIT = SQRT(RPI(1)*RPI(1) + RPI(2)*RPI(2) + RPI(3)*RPI(3) )       SIGH 640
      IF( ABS( RPIT-RANGE ).LT.1.0 ) GOTO 50                            SIGH 650
      RANGE = ABS(RPIT)                                                 SIGH 660
   30 CONTINUE                                                          SIGH 670
   50 CONTINUE                                                          SIGH 680
      P5TF2 = 0.5*TF*TF                                                 SIGH 690
C     TRANSFORM RELATIVE POSITION,VELOCITY AND PROJECTILE FUTURE POSITION SIGH 700
C     TO ATTACKER SYSTEM                                                SIGH 710
      DO   500   I = 1,3                                                SIGH 720
      AVELA(I) = 0.                                                     SIGH 730
      VTARG(I) = 0.                                                     SIGH 740
      ATARG(I) = 0.                                                     SIGH 750
      DO   500   J = 1,3                                                SIGH 760
      ATARG(I) = ATARG(I) + EULER(J,I) * ACCA(J)                        SIGH 770
      VTARG(I) = VTARG(I) + EULER(J,I) * VREL(J)                        SIGH 780
      AVELA(I) = AVELA(I) + EULER(J,I) * VELA(J)                        SIGH 790
  500 CONTINUE                                                          SIGH 800
```

```
      AVPRJ(1) = VELMUZ * COS(GUNALP) + AVELA(1)                        SIGH 810
      AVPRJ(2) = AVELA(2)                                               SIGH 820
      AVPRJ(3) = VELMUZ * SIN(GUNALP) + AVELA(3)                        SIGH 830
      DO    550    I = 1,3                                              SIGH 840
      PFP(I) = AVPRJ(I)*TF + PSTF2*GEE*EULER(3,I)                       SIGH 850
  550 CONTINUE                                                          SIGH 860
      TFPY = VTARG(2) * TF                                              SIGH 870
      TFPZ = VTARG(3) * TF                                              SIGH 880
      IF( IORD .EQ. 1 ) GO TO 600                                       SIGH 890
      TFPY = TFPY + ATARG(2)*PSTF2                                      SIGH 900
      TFPZ = TFPZ + ATARG(3)*PSTF2                                      SIGH 910
  600 CONTINUE                                                          SIGH 920
      BPY = PFP(2) - TFPY                                               SIGH 930
      BPZ = PFP(3) - TFPZ                                               SIGH 940
      HUDX = BPY/RANGE                                                  SIGH 950
      HUDY = -BPZ/RANGE                                                 SIGH 960
 5700 IF( NTRY.GT.00 ) RETURN                                           SIGH 970
 5701 DT = TIME-TOLD                                                    SIGH 980
      IF( DT.GE.1.E-4 ) GOTO 5600                                       SIGH 990
      HUDY = 0.0                                                        SIGH1000
      HUDX = 0.0                                                        SIGH1010
      HUDXD = 0.0                                                       SIGH1020
      HUDYD = 0.0                                                       SIGH1030
      GOTO 5610                                                         SIGH1040
 5600  CONTINUE                                                         SIGH1050
      HUDXD = ( HUDX - HUDXO ) / DT                                     SIGH1060
      HUDYD = ( HUDY - HUDYO ) / DT                                     SIGH1070
 5610 CONTINUE                                                          SIGH1080
      HUDXO = HUDX                                                      SIGH1090
      HUDYO = HUDY                                                      SIGH1100
      TOLD = TIME                                                       SIGH1110
      RETURN                                                            SIGH1120
 9999 CONTINUE                                                          SIGH1130
      RETURN                                                            SIGH1140
 5500 CALL SGHT(NTRY )                                                  SIGH1150
      RETURN                                                            SIGH1160
 7004 FORMAT(10X,' TARGET IS OUT OF RANGE AND PULLING AWAY.  RANGE = ', SIGH1170
     &            F15.5)                                                SIGH1180
 7001 FORMAT(10X,'ITERATIVE RANGE VECTOR = ',3F15.5)                    SIGH1190
 7000 FORMAT(10X,'TIME OF FLIGHT= ',F10.4,'RANGE = ',F10.4,             SIGH1200
```

```
      &'      UBAR = ',F15.5)                                          SIGH1210
    1 FORMAT(A6,I4,7E10.2)                                             SIGH1220
    2 FORMAT(1X,A6,I4,7E10.2)                                          SIGH1230
      END                                                              SIGH1240
```

```
      SUBROUTINE PROJ( ACOEF1,CDH1,CDH2,PSA3 )
      DIMENSION PMACH(5),PCD(5)
      DATA RHOZRO/ .0023769/,CONST1/ .0035683/,CONST2/518.688/
      DATA G/32.174/
      DATA AREAP/.00338/,WPROJ/1560./
      DATA PMACH/1.5,2.,3.,4.,5./
      DATA PCD/.322,.461,.369,.313,.293/
      PMACHI = 0.
      PMACH2 = 0.
      DO   10   I = 1,5
      PMACHI = PMACHI + (1./PMACH(I) )
   10 PMACH2 = PMACH2 + (1.0/PMACH(I) ) ** 2
      D = 1.0/(( 5. * PMACH2) - (PMACHI**2) )
      CDRAG1 = 0.
      CDRAG2 = 0.
      DO   20   I = 1,5
      CDRAG1 = CDRAG1 + ((PMACH2 - (PMACHI/PMACH(I))) * PCD(I))
   20 CDRAG2 = CDRAG2 + (((5./PMACH(I)) - PMACHI) * PCD(I) )
      CDH1 = D * CDRAG1
      CDH2 = D * CDRAG2
      VSONIC = 1117. - .004 * ABS( PSA3)
      CDH2 = CDH2 * VSONIC
      CONST3 = ABS(PSA3)
      ATTRHO = RHOZRO * (1.0 - (CONST1*CONST3)/CONST2)**4.2561
      ACOEF1 = 3499.96 * ATTRHO * G * AREAP / WPROJ
      RETURN
      END
```

```
      SUBROUTINE SGHT( NTRY )
      REAL KBRHO,KH,KSIG,LE,LA,LAD,LED
     &              ,LANEW,LENEW
      COMMON /GUN/ VM,GA,GHR,DA,DE
      COMMON /SIGHTS/ HALP,SIG
      COMMON /STATE/ GSV(80)
      COMMON/PILOTS/BODGIE(56)
      EQUIVALENCE    (GSV(31),RANGE),(BOOGIE(42),CV)
     &              ,(GSV(14),P),(GSV(15),Q),(GSV(16),R)
     &              ,(GSV(8),VA),(GSV(36),AN),(GSV(12),ALPHA)
     &               ,(GSV(1),TIME),(GSV(26),RHO)
     P              (GSV(77),HUDX),(GSV(78),HUDY),(GSV(79),HUDXD)
     &              ,(GSV(80),HUDYD)
C   PROGRAM DISRET         DISTURBED RETICLE SIGHT SIMULATION
C   RRH IS RECIPROCAL OF GUN HARMONIZATION RANGE(2000 FEET)
C   DA AND DE ARE Y AND Z DISTANCES  OF GUN FROM HUD IN FT
C   GA IS GUN ANGLE WITH RESPECT TO X AXIS OF AIRCRAFT
C   KBRHO IS A BALLISTIC COEFFICIENT DIVIDED BY SEA LEVEL AIR DENSITY
C   VM IS MUZZLE VELOCITY
C   RANGE IS IN FEET
C   VC IS NEGATIVE OF RANGE RATE IN FT/SEC
C   P,Q,R ARE ANGULAR RATES IN BODY AXES, RAD/SEC
C   AN IS NORMAL ACCELERATION IN FT/SEC(-32.17 WHEN STRAIGHT $ LEVEL)
C   ALPHA IS ANGLE OF ATTACK IN RADIANS
C   VA IS TRUE AIRSPEED IN FT/SEC
C   RHO IS AIR DENSITY IN SLUGS PER CUBIC FOOT
      DATA JUMP/0/
      IF( NTRY.GE.00 ) GOTO 2000
      LA = 0.0
      LE = 0.0
      TOLD = TIME
      RRH = 1./GHR
      COSGA = COS(GA)
      SINGA = SIN(GA)
C   SIG IS THE SIGHT DAMPING FACTOR
      KSIG = 1./( 1. + SIG )
 2000 DT = TIME - TOLD
      KBRHO = .00614/RHO
      VC = -CV
      RDC = 0.
```

```
      SQRV = SQRT( VA + VM )
      VLS = KSRHO * RHO * RANGE * SQRV
      VOS = VM + VC - VLS
      VCM = (VOS * VOS - 4. * (VA - VC) * VLS )
      VCM = SQRT( VCM )
      SLA = LA * (1. - .16667 * LA * LA )
      SLE = LE * (1. - .16667 * LE * LE )
      CLE = 1. - .5 * LE * LE
      CLA = 1. - .5 * LA * LA
      RCOS = 1. / (CLA*CLA )
      RDE = ( -VC-RDC*SLE ) * RCOS
      RE = RANGE * RCOS
C   APPROXIMATION USED FOR SQUARE ROOT
C   IF Y = APPROX. SQUARE ROOT OF X, THEN SQRT(X) = .5*(Y+X/Y) IS VERY CL
      SQRV = .5 * (SQRV + (VA+VM) / SQRV )
      VLS = KSRHO * RHO * RE * SQRV
      VOS = VM - RDE - VLS
C   APPROXIMATION USED FOR SQUARE ROOT
      VCM = .5 * (VCM + (VOS*VOS - 4. * (VA+RDE) * VLS ) / VCM)
      VE = .5 * (VOS+VCM)
C   RTF AND VF ARE 1/TIME OF FLIGHT AND AVG RELATIVE VELOCITY OF BULLET
      RTF = VE/RE
      VF = VE + RDE
      RRANGE = 1./RANGE
      KH = 1. - RANGE * RRH
      VN = VF - SIG * RDE * VLS * (VF+VA) / (VCM*VF)
      PG = P * COSGA - SINGA * R
      RG = P * SINGA + R * COSGA
      RDC = VA * (ALPHA+GA) * (VM-VF) / (VA+VM) + .5 * AN/RTF
C   WJ AND WK ARE COMPUTED SIGHT LINE ANGULAR RATES
      WK = (KH * DA * CLA * RTF - VN * SLA ) * RRANGE
      WJ = ( ( RDC - KH * DE * RTF ) * CLE - VN * CLA * SLE ) * RRANGE
      LED = (WJ + SLA * PG - CLA * Q ) * KSIG
      LAD = ( ( WK-SLE*(CLA*PG+SLA*Q) ) / CLE - RG ) * KSIG
      LA = LA + LAD*DT
      LE = LE + LED * DT
C   LA AND LE ARE AZIMUTH AND ELEVATION ANGLES OF COMPUTED SIGHT LINE
C   LA AND LE ARE IN RADIANS. COMPUTE PLA AND PLE AS NEGATIVE IN MILLS
      LANEW = LE + DT*LAD
      LENEW = LE + DT*LED
```

```
      HUDX = -LANEW                                                     DSRT 810
      HUDXD = -LAD                                                      DSRT 820
      HUDY = LENEW - HALP                                               DSRT 830
      HUDYD = LED                                                       DSRT 840
      IF( NTRY.GT.00 ) RETURN                                           DSRT 850
      LA = LANEW                                                        DSRT 860
      LE = LENEW                                                        DSRT 870
      TOLD = TIME                                                       DSRT 880
      RETURN                                                            DSRT 890
    1 FORMAT(A6,4X 7E10.2)                                              DSRT 900
      END                                                               DSRT 910
```

```
      SUBROUTINE ESCAPE(   NTRY  )
      DIMENSION ICAL(25,10),RCAL(25,10),TSV(30),TSVOLD(6)
     &           ,TSYSV(80)
     &           ,TUV(3,3),TEUA(3),TX(3),TV(3),ICVL(25)
     &   ,RCVL(25),F(5),NQL(10),NQU(10),STVLL(10),STVUL(10)
     &     ,RTUV(3,3)
     &       ,AK(3),AT(3)
     &         ,FTSV(3)
      COMMON /STATE/ GSV(80)
      EQUIVALENCE (ICVL(2),NLWR),(ICVL(3),NQL),(ICVL(13),NUPR)
     &      ,(TSYSV(1),GSV(1))
     &    ,(ICVL(14),NQU),(RCVL(1),F(1)),(RCVL(6),STVLL(1))
     &    ,(RCVL(16),STVUL(1))
     &    ,(TSV(1),TX(1)),(TSV(4),TV(1)),(TSV(7),V)
     &    ,(GSV(32),TSV(1)),(GSV(1),T),(TSV(13),TEUA(1))
     &                     ,(AK(1),TSV(8)),(AN,TSV(11))
     &     , (TAA,TSV(16) )
     &   ,(TSV(12),AV)
      DATA G/32.2/
      DATA JUMP/0/
      IF( NTRY.GE.00 ) GOTO 250
      IF( JUMP.NE.00 ) GOTO 51
      DUM = TAAF(0.,0.,0.)
      DUM = FAN(0.,0.)
      DUM = FAVD(0.,0.,0.,0.)
      DO    50  I=1,10
      DO    50      J=1,25
      ICAL(J,I) = 00
 50   RCAL(J,I) = 0.0
 51   READ1,L,JMX
      PRINT 2,L,JMX
      IF( JMX.EQ.00 ) GOTO 200
      NMNVRS = JMX
      DO       100          I=1  ,NMNVRS
      READ 1,L,ICAL(1,I),(RCAL(J,I),J=1,5)
      PRINT 2,L,ICAL(1,I),(RCAL(J,I),J=1,5)
      READ 3,L,JMX,(ICAL(J+2,I),RCAL(J+5,I),J=1,JMX)
      ICAL(2,I) = JMX
      PRINT 4,L,JMX,(ICAL(J+2,I),RCAL(J+5,I),J=1,JMX)
      READ 3,L,JMX,(ICAL(J+13,I),RCAL(J+15,I),J=1,JMX)
```

```
      ICAL(13,I) = JMX
      PRINT 4,L,JMX,(ICAL(J+13,I),RCAL(J+15,I),J=1,JMX)
100   CONTINUE
200   READ 1,L,MNVR,(TSV(J),J=1,6)
      PRINT2,L,MNVR,(TSV(J),J=1,6)
      DO       210               J=1,6
210   TSVOLD(J) = TSV(J)
      DO       220               J=1,3
220   AK(J) = 0.0
      TOLD = T
250   IF( MNVR.GT.NMNVRS ) GOTO 900
300   VS = 0.0
      DLT = T-TOLD
      DO       310               J=1,3
      TSV(J+3) = TSVOLD(J+3) + AK(J)*DLT
      VS = VS + TSV(J+3) **2
      TSV(J) = TSVOLD(J) + 0.5*( TSV(J+3)+TSVOLD(J+3) )*DLT
310   AT(J) = AK(J)
      AT(3) = AT(3) - G
      V = SQRT( VS )
      TSV(7) = V
      VM = V/VSND( -TSV(3) )
      ASSIGN 345 TO KIKBAK
315   ANS = 0.0
      ATS = 0.0
      DO       320               J=1,3
      ATS = ATS + AT(J)**2
      L = MOD(J,3)+1
      M = MOD(L,3)+1
      TUV(J,2) = TV(L)*AT(M) - TV(M)*AT(L)
320   ANS = ANS + TUV(J,2)**2
      IF( ANS.NE.0.0 ) GOTO 325
      ANS = VS
      TUV(3,2) = V
325   DUM = SQRT( ANS )
      DO       330 J=1,3
      TUV(J,1) = TV(J) / V
330   TUV(J,2) = TUV(J,2)/DUM
      DO    340    J = 1,3
      L = MOD(J,3) + 1
```

```
      M = MOD(L,3) + 1
  340 TUV(J,3) = TUV(L,1) * TUV(M,2) - TUV(M,1) * TUV(L,2)
      AVS = 0.0
      ANS = 0.0
      DO      341     J=1,3
       AVS = AVS + AT(J)*TUV(J,1)
 341   ANS = ANS + AT(J)*TUV(J,3)
       AV = ABS( AVS )
       AN = ABS( ANS )
       GOTO KIKBAK,( 345,600 )
  345 TAA = TAAF( AN,VM,-TX(3) )
      SAA = SIN( TAA*.0174533 )
      CAA = COS( TAA*.0174533 )
      DO    350    J = 1,3
      RTUV(J,1) = CAA * TUV(J,1) - SAA * TUV(J,3)
      RTUV(J,3) = SAA * TUV(J,1) + CAA * TUV(J,3)
  350 RTUV(J,2) = TUV(J,2)
      IF( ABS( RTUV(3,1) ).GT.1.0 ) RTUV(3,1) = SIGN( 1.0,RTUV(3,1) )
      TEUA(1)  = ATAN2( RTUV(3,2),RTUV(3,3) )*57.2958
      TEUA(2) = ARSIN( -RTUV(3,1) )*57.2958
      TEUA(3)  = ATAN2( RTUV(2,1),RTUV(1,1) )*57.2958
      IF( NTRY .GT. 00 ) RETURN
      IF( NTRY .LT. 00 ) GO TO 400
      TOLD = T
      DO           355        J=1,6
 355  TSVOLD(J) = TSV(J)
      DO    360    J = 1,NLWR
      IF( TSYSV(NOL(J) ) .LE. STVLL(J) ) GO TO 400
  360 CONTINUE
      DO    370    J = 1,NUPR
      IF( TSYSV( NOU(J) ) .GE. STVUL(J) ) GO TO 400
  370 CONTINUE
      GO TO 500
  400 MNVR = MNVR + 01
      IF( MNVR.GT.NMNVRS ) GOTO 900
      KIK = 00
      DO    410    J = 1,25
      ICVL(J) = ICAL(J,MNVR)
      RCVL(J) = RCAL(J,MNVR)
 410  CONTINUE
```

```
      IZG = ICVL(1)                                                     ESCP1210
      IF( IZG.GT.00 ) GOTO 440                                          ESCP1220
      DO    420       J=1,3                                             ESCP1230
  420 FTSV(J) = F(J)                                                    ESCP1240
      DO    430     I=1,3                                               ESCP1250
      F(I) = 0.0                                                        ESCP1260
      DO          430       J=1,3                                       ESCP1270
  430 F(I) = F(I) + TUV(I,J)*FTSV(J)                                    ESCP1280
  440 IZG = IABS( IZG )                                                 ESCP1290
      IHOP = 00                                                         ESCP1300
      IF( IZG .GT. 3 )  IHOP = 3                                        ESCP1310
  500  GOTO    ( 510,520,530,510,520,530,590,560 ), IZG                 ESCP1320
  510   IZG = 08                                                        ESCP1330
      DO          511         J=1,3                                     ESCP1340
  511 FTSV(J) = F(J)                                                    ESCP1350
      GOTO   550                                                        ESCP1360
  520  IZG = 08                                                         ESCP1370
  530  DO          540          J=1,3                                   ESCP1380
  540  FTSV(J) = F(J) + TSVOLD(J+IHOP)                                  ESCP1390
  550 TFUT = TOLD + F(4)                                                ESCP1400
  560 DLT = TFUT - TOLD                                                 ESCP1410
      IF( IHOP .NE. 0 ) GO TO 570                                       ESCP1420
      DLT2 = DLT * DLT                                                  ESCP1430
      DO    565    J = 1,3                                              ESCP1440
  565 AK(J) = 2. * ( FTSV(J) - TSVOLD(J) ) / DLT2                       ESCP1450
      GO TO 580                                                         ESCP1460
  570 DO    575    J = 1,3                                              ESCP1470
  575 AK(J) = ( FTSV(J) - TSVOLD (J+3) )/DLT                            ESCP1480
  580 DO    585    J = 1,3                                              ESCP1490
  585 AT(J) = AK(J)                                                     ESCP1500
      AT(3) = AT(3) - G                                                 ESCP1510
       ASSIGN 600 TO KIKBAK                                             ESCP1520
      GO TO 315                                                         ESCP1530
  590 IF( KIK .NE. 0 ) GO TO 592                                        ESCP1540
      IF( F(4).EQ.0.0 ) GOTO 591                                        ESCP1550
      F(2) = ( F(2)-F(1) )/F(4)                                         ESCP1560
  591   TREF = TOLD                                                     ESCP1570
  592 RAD = F(1) + F(2) * ( TOLD - TREF )                               ESCP1580
      DO          593       J=1,3                                       ESCP1590
  593 TUV(J,1) = TV(J)/V                                                ESCP1600
```

BEST AVAILABLE COPY

```
      DUM = 1.0-TUV(3,1)*TUV(3,1)                                       ESCP1610
      IF( DUM.LT..0001 ) DUM = .0001                                    ESCP1620
      TUV(3,3) = SQRT( DUM )                                            ESCP1630
      TUV(2,2) = TUV(1,1)/TUV(3,3)                                      ESCP1640
      TUV(1,3) = -TUV(2,2)*TUV(3,1)                                     ESCP1650
      TUV(1,2) = -TUV(2,1)/TUV(3,3)                                     ESCP1660
      TUV(2,3) = TUV(1,2)*TUV(3,1)                                      ESCP1670
      TUV(3,2) = 0.0                                                    ESCP1680
      PHI = RAD*.0174533                                                ESCP1690
      CPH = COS(PHI)                                                    ESCP1700
      SPH = SIN( PHI )                                                  ESCP1710
      DO   595   J=2,3                                                  ESCP1720
      DUM = TUV(J,2)*CPH + TUV(J,3)*SPH                                 ESCP1730
      TUV(J,3) = -TUV(J,2)*SPH + TUV(J,3)*CPH                           ESCP1740
  595 TUV(J,2) = DUM                                                    ESCP1750
      AN = F(3) * G                                                     ESCP1760
       AV = FAVD( AN,VM,-TSVOLD(3),F(5) )                               ESCP1770
      AVS = AV                                                          ESCP1780
  600 ANMX = FAN( VM,-TSVOLD(3) )                                       ESCP1790
      IF( AN .GT. ANMX ) AN = ANMX                                      ESCP1800
      AVMX = FAVD( AN,VM,-TSVOLD(3),F(5) )                              ESCP1810
      AV = SIGN( AVS,AV )                                               ESCP1820
      IF( AVMX .LT. AV ) AV = AVMX                                      ESCP1830
      DO   610   J = 1,3                                                ESCP1840
  610 AK(J) = AV * TUV(J,1) - AN * TUV(J,3)                             ESCP1850
      AK(3) = AK(3) + G                                                 ESCP1860
      KIK = KIK + 01                                                    ESCP1870
      RETURN                                                            ESCP1880
  900 DO      910    I=1,6                                              ESCP1890
  910 TSV(I) = 1.E75                                                    ESCP1900
      DO          920        I=7,30                                     ESCP1910
  920 TSV(I) = 0.0                                                      ESCP1920
      PRINT5                                                            ESCP1930
      RETURN                                                            ESCP1940
    1 FORMAT(A6,I4,7E10.2)                                              ESCP1950
    2 FORMAT( 1X A7,I4,1P 7E12.3 )                                      ESCP1960
    3 FORMAT(A6,I4,4(I5,E10.2)/(10X4(I5,E10.2)))                        ESCP1970
    4 FORMAT( 1X A7,I4,1P 4( I5,E12.3 )/( 11X 4( I5,E12.3 ) ) )         ESCP1980
    5 FORMAT(13H0***... PCOF )                                          ESCP1990
      END                                                               ESCP2000
```

```
      FUNCTION VSND( H )
      VSND = 968.12
      IF( H.GE.36000. ) RETURN
      VSND = 49.*SQRT(518.7-H*3.565E-3)
      RETURN
      END
```

```
      FUNCTION FAN( VM,H )
      DIMENSION AM(3),AH(3),FMH(9)
      DATA NM,NH/3,3/
      DATA JUMP/0/
      IF( JUMP.GT.00 ) GOTO 100
      JUMP = 01
      READ1,LABEL,AM
      PRINT2,LABEL, AM
      READ1,LABEL,AH
      PRINT2,LABEL,AH
      READ1,LABEL,FMH
      PRINT2,LABEL,FMH
    1 FORMAT(A4,6X 7F10.2/(10X7E10.2))
    2 FORMAT(1XA10,1P 7E12.3/(11X7E12.3))
      FAN = 0.
      RETURN
  100 CALL LOOK2( DAFT,VM,H,AM,NM,AH,NH,FMH,0 )
      FAN = DAFT
      RETURN
      END
```

```
      FUNCTION FAVD(AN,VM,H,TP )                                        FAND  10
      DIMENSION AAN(3),AM(3),AH(3),TRST(9),DRG(27)                      FAND  20
      DATA NA,NM,NH/3,3,3/                                              FAND  30
      DATA JUMP/0/                                                      FAND  40
      IF( JUMP.GT.00 ) GOTO 100                                         FAND  50
      JUMP = 1                                                          FAND  60
      READ1,LABEL,AAN                                                   FAND  70
      PRINT2,LABEL,AAN                                                  FAND  80
      READ1,LABEL,AM                                                    FAND  90
      PRINT2,LABEL,AM                                                   FAND 100
      READ1,LABEL,AH                                                    FAND 110
      PRINT2,LABEL,AH                                                   FAND 120
      READ1,LABEL,TRST                                                  FAND 130
      PRINT2,LABEL,TRST                                                 FAND 140
      READ1,LABEL,DRG                                                   FAND 150
      PRINT2,LABEL,DRG                                                  FAND 160
    1 FORMAT(A4,6X 7E10.2/(10X7E10.2))                                  FAND 170
    2 FORMAT(1XA10,1P 7E12.3/(11X7E12.3))                               FAND 180
      FAVD = 0.                                                         FAND 190
      RETURN                                                            FAND 200
  100 CALL LOOK3( DRAG,AN,VM,H,AAN,NA,AM,NM,AH,NH,DRG,0 )               FAND 210
      CALL LOOK2( THRST,VM,H,AM,NM,AH,NH,TRST,0 )                       FAND 220
      FAVD = THRST*TP - DRAG*AN                                         FAND 230
      RETURN                                                            FAND 240
      END                                                               FAND 250
```

```
      FUNCTION TAAF( AN,VM,H )
      DIMENSION TAN(3),TM(3),TH(3),TAA(27)
      DATA NA,NM,NH/3,3,3/
      DATA JUMP/0/
      IF( JUMP.GT.00 ) GOTO 50
      JUMP = 01
      READ1,LABEL,TAN
      PRINT2,LABEL,TAN
      READ1,LABEL,TM
      PRINT2,LABEL,TM
      READ1,LABEL,TH
      PRINT2,LABEL,TH
      READ1,LABEL,TAA
      PRINT2,LABEL,TAA
    1 FORMAT(A4,6X 7E10.2/(10X7E10.2))
    2 FORMAT(1XA10,1P 7E12.3/(11X7E12.3))
      TAAF = 0.
      RETURN
   50 CALL LOOK3( DAFT,AN,VM,H,TAN,NA,TM,NM,TH,NH,TAA,0 )
      TAAF = DAFT
      RETURN
      END
```

```
      SUBROUTINE LOOK1( OUT,X,AX,NX,FX,NCX )                           LUK1  10
      DIMENSION AX(NX),FX(NX)                                          LUK1  20
      IF( NCX.GT.00 ) GOTO 400                                         LUK1  30
      IF( X.GT.AX(1) ) GOTO 100                                        LUK1  40
      I = 02                                                           LUK1  50
      B = 0.                                                           LUK1  60
      A = 1.                                                           LUK1  70
      GOTO     400                                                     LUK1  80
  100 DO       200          I=2,NX                                     LUK1  90
      IF( X.LE.AX(I) ) GOTO 300                                        LUK1 100
  200 CONTINUE                                                         LUK1 110
      I = NX                                                           LUK1 120
      B = 1.0                                                          LUK1 130
      A = 0.                                                           LUK1 140
      GOTO     400                                                     LUK1 150
  300 B = AX(I)-AX(I-1)                                                LUK1 160
      A = ( AX(I)-X )/B                                                LUK1 170
      B = ( X-AX(I-1) )/B                                              LUK1 180
  400 OUT = A*FX(I-1) + B*FX(I)                                        LUK1 190
      RETURN                                                           LUK1 200
      END                                                              LUK1 210
```

```
      SUBROUTINE LOOK2( OUT,X,Y,AX,NX,AY,NY,FXY,NCXY )                  LUK2  10
      DIMENSION AX(NX),AY(NY),FXY(NX,NY)                                LUK2  20
      IF( NCXY.GT.00 ) GOTO 400                                         LUK2  30
      NCX = 00                                                          LUK2  40
      IF( Y.GT.AY(1) ) GOTO 100                                         LUK2  50
      I = 2                                                             LUK2  60
      B = 0.0                                                           LUK2  70
      A = 1.0                                                           LUK2  80
      GOTO 400                                                          LUK2  90
  100 DO      200      I=2,NY                                           LUK2 100
      IF( Y.LE.AY(I) ) GOTO 300                                         LUK2 110
  200 CONTINUE                                                          LUK2 120
      I = NY                                                            LUK2 130
      A = 0.0                                                           LUK2 140
      B = 1.0                                                           LUK2 150
      GOTO 400                                                          LUK2 160
  300 B = AY(I)-AY(I-1)                                                 LUK2 170
      A = ( AY(I)-Y )/B                                                 LUK2 180
      B = ( Y-AY(I-1) )/B                                               LUK2 190
  400 CALL LOOK1( FA,X,AX,NX,FXY(1,I-1),NCX)                            LUK2 200
      NCX = 01                                                          LUK2 210
      CALL LOOK1( FB,X,AX,NX,FXY(1,I),NCX)                              LUK2 220
      OUT = FA*A + FB*B                                                 LUK2 230
      RETURN                                                            LUK2 240
      END                                                               LUK2 250
```

```
      SUBROUTINE LOOK3( OUT,X,Y,Z,AX,NX,AY,NY,AZ,NZ,FXYZ,NCXYZ )        LUK3  10
      DIMENSION AX(NX),AY(NY),AZ(NZ),FXYZ(NX,NY,NZ)                     LUK3  20
      IF( NCXYZ.GT.00 ) GOTO 400                                        LUK3  30
      NXNY = NX*NY                                                      LUK3  40
      NCXY = 00                                                         LUK3  50
      IF( Z.GT.AZ(1) ) GOTO 100                                         LUK3  60
      I = 02                                                            LUK3  70
      A = 1.0                                                           LUK3  80
      B = 0.0                                                           LUK3  90
      GOTO 399                                                          LUK3 100
  100 DO       200          I=2,NZ                                      LUK3 110
      IF( Z.LE.AZ(I) ) GOTO 300                                         LUK3 120
  200 CONTINUE                                                          LUK3 130
      I = NZ                                                            LUK3 140
      A = 0.0                                                           LUK3 150
      B = 1.0                                                           LUK3 160
      GOTO 399                                                          LUK3 170
  300 B =  AZ(I)-AZ(I-1)                                                LUK3 180
      A = ( AZ(I)-Z )/B                                                 LUK3 190
      B = ( Z-AZ(I-1) )/B                                               LUK3 200
  399  CONTINUE                                                         LUK3 210
  400 CALL LOOK2( FA,X,Y,AX,NX,AY,NY,FXYZ(1,1,I-1),NCXY)                LUK3 220
      NCXY = 01                                                         LUK3 230
      CALL LOOK2( FB,X,Y,AX,NX,AY,NY,FXYZ(1,1,I),NCXY)                  LUK3 240
      OUT = FA*A + FB*B                                                 LUK3 250
      RETURN                                                            LUK3 260
      END                                                               LUK3 270
```

	ATTACKER		TARGET
1	x_1	$\dot{x}_1$	x_1
2	x_2	$\dot{x}_2$	x_2
3	x_3	$\dot{x}_3$	x_3
4	v_1	$\dot{v}_1$	v_1
5	v_2	$\dot{v}_2$	v_2
6	v_3	$\dot{v}_3$	v_3
7	v	$\dot{v}$	v
8	ϕ	$\dot{\phi}$	A_{k1}
9	θ	$\dot{\theta}$	A_{k2}
10	ψ	$\dot{\psi}$	A_{k3}
11	α	$\dot{\alpha}$	A_n
12	β	$\dot{\beta}$	A_t
13	P	$\dot{P}$	ψ degrees
14	Q	$\dot{Q}$	θ degrees
15	R	$\dot{R}$	ϕ degrees

```
*****************************************************************************************
ELAPSED TIME      0.0
   1        0.0               7.0000E+02                           2.0000E+03
   2        0.0               0.0                                  1.0000E+03
   3     -2.0000E+04          0.0                                 -2.1000E+04
   4      7.0000E+02          2.9416E+01                           7.0000E+02
   5        0.0              -1.0154E+00                           0.0
   6        0.0               2.6573E+01                           0.0
   7      7.0000E+02          2.9416E+01                           7.0000E+02
   8        0.0               0.0                                  1.7570E+01
   9        0.0               0.0                                  1.2800E+01
  10        0.0               0.0                                  1.9200E+01
  11        0.0               3.7962E-02                           1.8244E+01
  12        0.0              -1.4506E-03                           1.7570E+01
  13        0.0               4.9865E+00                           4.4606E+01
  14        0.0               1.1569E+00                           2.4237E+00
  15        0.0               1.5278E-01                           2.3885E+00
CONTROL VECTOR =         5.2360E-01        -1.0845E-01        1.3700E+04       0.0
ERROR VECTOR =         2.0000E+03         1.0000E+03        -1.0000E+03
     BANK               YANK               RANGE               SIGMA              ERROR
   4.5332E-01         4.1054E-01         2.4495E+03          4.6365E-01          6.1928E-01
   0.0                0.0                0.0                 0.0                 0.0
  -4.9590E-01         1.1987E+00        -1.0210E+00          0.0                 7.9468E-01
PIPPER     0.0                0.0                   4.4899E-02         0.0
*****************************************************************************************
```

```
ELAPSED TIME      2.100
     1      1.4777E+03      6.7397E+02                     3.5083E+03
     2      1.6955E+02      2.1299E+02                     1.0310E+03
     3     -2.0108E+04     -1.2336E+02                    -2.0954E+04
     4      6.7397E+02     -5.4567E+00                     7.3619E+02
     5      2.1299E+02      6.1561E+01                     3.1011E+01
     6     -1.2336E+02      3.0237E+00                     4.6516E+01
     7      7.4712E+02      1.9343E+01                     7.3831E+02
     8      1.0437E+00     -9.1589E-01                     1.6615E+01
     9      2.2939E-01     -4.6674E-01                     1.7204E+01
    10      5.6019E-01     -2.4631E-01                     2.5806E+01
    11      9.1255E-02     -5.7762E-01                     1.8104E+01
    12     -8.9906E-02     -3.0952E-01                     1.6887E+01
    13     -8.5988E-01     -1.9110E+00                     6.5464E+01
    14     -4.4209E-01     -7.7502E-01                    -2.2223E+00
    15      2.8272E-01     -8.6246E-01                     5.4709E+00
CONTROL VECTOR =      -5.2360E-01      4.3382E-02      1.3700E+04      0.0
ERROR VECTOR =         2.3131E+03     -4.5909E+02      1.3686E+02
       BANK            YANK            RANGE           SIGMA           ERROR
   -3.3330E-01     -8.4574E-02      2.3622E+03     -1.6384E-01      1.8438E-01
   -2.9625E-01      3.0627E-01     -7.3673E+01     -2.6304E-01      9.3253E-02
    1.2142E+00      1.3409E+00      2.0223E+01     -5.6170E-02      2.7168E-01
PIPPER      -3.2082E-02     -5.8857E-02      6.1510E-02      1.9118E-01
*******************************************************************************
ELAPSED TIME      2.200
     1      1.5451E+03      6.7451E+02                     3.5820E+03
     2      1.9111E+02      2.1751E+02                     1.0341E+03
     3     -2.0120E+04     -1.2186E+02                    -2.0949E+04
     4      6.7451E+02      5.5252E+00                     7.3785E+02
     5      2.1751E+02      4.3637E+01                     3.2731E+01
     6     -1.2186E+02      1.2690E+01                     4.9096E+01
     7      7.4905E+02      1.9493E+01                     7.4020E+02
     8      9.4354E-01     -9.6992E-01                     1.6528E+01
     9      1.8592E-01     -3.3808E-01                     1.7456E+01
    10      5.3581E-01     -1.4553E-01                     2.6184E+01
    11      3.5192E-02     -4.4658E-01                     1.8172E+01
    12     -1.1410E-01     -1.7349E-01                     1.6848E+01
    13     -9.4302E-01      9.0764E+00                     6.6662E+01
    14     -3.1423E-01     -1.7374E+00                    -2.4787E+00
    15      1.8978E-01     -6.2726E-01                     5.6260E+00
CONTROL VECTOR =       5.2360E-01      1.3963E-01      1.3700E+04      0.0
ERROR VECTOR =         2.2974E+03     -5.1740E+02      1.4053E+01
       BANK            YANK            RANGE           SIGMA           ERROR
   -3.4362E-01     -5.4827E-02      2.3550E+03     -1.8793E-01      1.9577E-01
   -1.9303E-01      1.8288E-01     -7.1497E+01     -1.7980E-01      1.2139E-01
   -6.7944E-02     -1.5343E-01      2.1842E+01     -8.0797E-02      3.8124E-01
PIPPER      -3.3583E-02     -1.4804E-02      8.3763E-02      2.2490E-01
*******************************************************************************
```

```
ELAPSED TIME      7.200
     1        5.2877E+03      8.2590E+02                          7.4399E+03
     2        9.0558E+02      1.2597E+02                          1.4811E+03
     3       -2.0522E+04      1.2991E+02                         -2.0278E+04
     4        8.2590E+02      1.6750E+01                          7.9140E+02
     5        1.2597E+02      5.4855E+01                          1.6281E+02
     6        1.2991E+02      1.6096E+01                          2.4422E+02
     7        8.4752E+02      3.1479E+01                          8.4408E+02
     8       -5.1201E+00     -2.5672E+00                         -1.9027E+00
     9       -1.3822E-01      5.0685E-02                          3.7976E+01
    10        3.2188E-01     -2.4574E-02                          5.6962E+01
    11        5.6871E-02     -1.7313E-01                          4.3560E+01
    12       -4.7920E-02     -5.4028E-02                          1.2706E+01
    13       -2.5706E+00     -1.7404E+00                          1.1927E+02
    14       -2.2460E-03     -1.7932E-01                         -1.8789E+01
    15       -5.6182E-02      6.5184E-01                          1.5234E+01
CONTROL VECTOR =       -5.2355E-01    -1.5586E-02     1.3700E+04     -2.6180E-01
ERROR VECTOR =          2.2362E+03    -1.1278E+02     9.8199E+01
       BANK             YANK          RANGE           SIGMA          ERROR
  -1.1564E-03      -9.1291E-03     2.2412E+03     -5.6362E-04     9.1465E-03
   1.5995E-01      -1.7990E-02    -1.1213E+01      7.7977E-02     1.3151E-02
   1.1484E-02       1.1512E-01    -1.0886E+01     -3.2019E-04     5.6637E-01
PIPPER      -4.9830E-02     -7.9131E-02      2.0307E-04      1.4774E-01
GOTCHA
*******************************************************************************
ELAPSED TIME      7.300
     1        5.3703E+03      8.2769E+02                          7.5190E+03
     2        9.1843E+02      1.3090E+02                          1.4976E+03
     3       -2.0509E+04      1.3152E+02                         -2.0253E+04
     4        8.2769E+02      2.2477E+01                          7.9121E+02
     5        1.3090E+02      3.6808E+01                          1.6661E+02
     6        1.3152E+02      8.3396E+00                          2.4991E+02
     7        8.5065E+02      3.1810E+01                          8.4631E+02
     8       -5.3831E+00     -2.6818E+00                         -2.8389E+00
     9       -1.3459E-01      3.0891E-02                          3.8590E+01
    10        3.1848E-01     -4.6233E-02                          5.7884E+01
    11        3.8918E-02     -1.8067E-01                          4.4721E+01
    12       -5.1863E-02     -2.6027E-02                          1.2528E+01
    13       -2.6880E+00     -2.4534E+00                          1.1983E+02
    14       -1.6689E-02     -5.5489E-02                         -1.9210E+01
    15       -5.2676E-02      1.5216E+00                          1.5541E+01
CONTROL VECTOR =       -5.2355E-01    -1.5586E-02     1.3700E+04     -5.2360E-01
ERROR VECTOR =          2.2362E+03    -1.1136E+02     6.8372E+01
       BANK             YANK          RANGE           SIGMA          ERROR
   1.9263E-02      -1.3868E-02     2.2400E+03      9.6454E-03     1.6892E-02
   2.4439E-01      -4.7981E-02    -1.2239E+01      1.2148E-01     1.0875E-01
   4.3027E-02      -4.7303E-02    -1.0812E+01     -1.6689E-03     3.4759E-01
PIPPER      -5.9402E-02     -9.6455E-02      1.8244E-02      1.8059E-01
*******************************************************************************
```

```
ELAPSED TIME      9.400
     1      7.1506E+03      8.6723E+02                      9.1569E+03
     2      1.2501E+03      1.8012E+02                      1.9420E+03
     3     -2.0135E+04      2.2684E+02                     -1.9587E+04
     4      8.6723E+02      2.2490E+01                      7.5782E+02
     5      1.8012E+02     -6.7714E+00                      2.6178E+02
     6      2.2684E+02      3.8367E+01                      3.9267E+02
     7      9.1726E+02      2.7241E+01                      8.9275E+02
     8     -9.1426E+00     -2.0926E+00                     -3.5241E+01
     9     -2.1365E-01     -2.9417E-01                      5.3148E+01
    10      2.2445E-01     -2.9391E-01                      7.9720E+01
    11      1.8429E-02      1.7529E-01                      7.9257E+01
    12     -6.8848E-02     -2.2951E-01                      6.5719E+00
    13     -2.1549E+00     -1.5067E+00                      1.2644E+02
    14      3.6251E-01      5.1182E+00                     -2.9583E+01
    15      1.9398E-01     -2.2235E-01                      2.4170E+01
CONTROL VECTOR =       -5.2360E-01     -3.4907E-01      6.8500E+03      0.0
ERROR VECTOR =          2.1783E+03     -2.4365E+02     -2.1387E+01
        BANK                YANK            RANGE           SIGMA            ERROR
   -2.8214E-01          3.2951E-02      2.1920E+03     -1.5681E-01      1.6023E-01
   -1.9845E-01          1.4491E-01     -3.2889E+01     -1.5838E-01      1.8479E-01
   -1.9824E+00         -4.2191E+00     -3.0277E+00     -1.5222E-02     -7.7824E-01
PIPPER      4.5418E-02      -8.5070E-02       1.1710E-02      -1.9946E-01
**************************************************************************************

ELAPSED TIME      9.500
     1      7.2374E+03      8.6894E+02                      9.2325E+03
     2      1.2680E+03      1.7884E+02                      1.9685E+03
     3     -2.0112E+04      2.3362E+02                     -1.9547E+04
     4      8.6894E+02      1.5433E+01                      7.5429E+02
     5      1.7884E+02     -1.4783E+01                      2.6710E+02
     6      2.3362E+02      7.4990E+01                      4.0064E+02
     7      9.1997E+02      2.6679E+01                      8.9488E+02
     8     -9.3255E+00     -1.6916E+00                     -3.7619E+01
     9     -2.5491E-01     -4.3008E-01                      5.3870E+01
    10      1.9610E-01     -2.3993E-01                      8.0803E+01
    11      4.1818E-02      2.0930E-01                      8.1497E+01
    12     -9.3243E-02     -2.4729E-01                      6.1294E+00
    13     -1.7521E+00     -1.4819E+00                      1.2651E+02
    14      4.5097E-01      1.8304E+00                     -3.0161E+01
    15      1.8843E-01     -3.7888E-01                      2.4727E+01
CONTROL VECTOR =       -5.2360E-01     -1.7453E-01      6.8500E+03      0.0
ERROR VECTOR =          2.1682E+03     -2.9868E+02      1.0888E+01
        BANK                YANK            RANGE           SIGMA            ERROR
   -3.1400E-01          4.0730E-02      2.1887E+03     -1.7089E-01      1.7568E-01
   -1.9256E-01          9.7358E-02     -3.3161E+01     -1.4363E-01      1.6229E-01
   -4.8812E-01         -1.[illegible]329E+00     -3.0817E+00     -3.9910E-03     -2.1412E-01
PIPPER      3.3995E-02      -1.1414E-01      -1.0801E-02      -2.2561E-01
**************************************************************************************
```

```
ELAPSED TIME     11.100
   1      8.6002E+03      8.2357E+02                        1.0373E+04
   2      1.5992E+03      2.7492E+02                        2.4690E+03
   3     -1.9585E+04      3.8786E+02                       -1.8796E+04
   4      8.2357E+02     -2.7387E+01                        6.5770E+02
   5      2.7492E+02      9.8240E+01                        3.6101E+02
   6      3.8786E+02     -6.9629E-01                        5.4150E+02
   7      9.5713E+02      1.3156E+01                        9.2527E+02
   8     -1.1155E+01     -3.2599E-01                       -9.2237E+01
   9     -2.4225E-01      5.7703E-02                        6.2921E+01
  10      4.9321E-01     -1.5126E-01                        9.4378E+01
  11      9.5440E-02     -1.9519E-01                        1.2772E+02
  12      1.2646E-01      5.3909E-02                        4.6262E+00
  13     -3.6227E-01      2.7957E+00                        1.2325E+02
  14     -1.3585E-01     -6.6170E-01                       -4.0915E+01
  15     -8.0234E-02      1.2952E+00                        3.7679E+01
CONTROL VECTOR =       5.2360E-01      -8.7175E-03      -1.3050E+04      0.0
ERROR VECTOR =         2.1051E+03       2.7652E+02       1.1874E+02
        BANK              YANK             RANGE            SIGMA            ERROR
    2.9212E-01        3.3067E-02       2.1266E+03       1.4294E-01       1.4672E-01
    3.3421E-01        4.5009E-02      -4.6148E+01       1.8676E-01       1.9210E-01
   -5.4949E-01        9.2016E-01      -6.7562E+00       5.9527E-02       2.6540E-01
PIPPER      -1.2337E-02      -3.0688E-02      -5.4030E-02      1.0006E-01
*********************************************************************************

ELAPSED TIME     11.200
   1      8.6824E+03      8.2125E+02                        1.0439E+04
   2      1.6271E+03      2.8368E+02                        2.5054E+03
   3     -1.9546E+04      3.8781E+02                       -1.8742E+04
   4      8.2125E+02     -1.6475E+01                        6.4848E+02
   5      2.8368E+02      6.9438E+01                        3.6730E+02
   6      3.8781E+02      2.0070E+00                        5.5094E+02
   7      9.5845E+02      1.3335E+01                        9.2680E+02
   8     -1.1173E+01     -4.3535E-02                       -9.6898E+01
   9     -2.4286E-01     -6.9123E-02                        6.3150E+01
  10      4.8661E-01     -9.6122E-02                        9.4722E+01
  11      7.6662E-02     -1.6933E-01                        1.3136E+02
  12      1.2686E-01     -4.7720E-02                        5.6058E+00
  13     -6.6651E-02      3.2142E+00                        1.2269E+02
  14     -1.0406E-01      2.5834E+00                       -4.1661E+01
  15      5.1528E-02      1.3285E+00                        3.8826E+01
CONTROL VECTOR =       5.2360E-01      -1.7453E-01      -1.3050E+04      0.0
ERROR VECTOR =         2.0991E+03       2.9765E+02       8.7270E+01
        BANK              YANK             RANGE            SIGMA            ERROR
    3.1112E-01        3.9200E-02       2.1219E+03       1.5759E-01       1.6239E-01
    1.3448E-01        6.2372E-02      -4.6574E+01       9.2777E-02       1.0509E-01
    1.2643E+00       -2.1104E+00      -5.1524E+00       2.9846E-02      -4.7151E-01
PIPPER      -1.6730E-02      -4.3833E-02      -4.5435E-02      8.6385E-02
*********************************************************************************
```


(The reverse of this page is blank)

INITIAL DISTRIBUTION

AFATL/DLYD	12
ASD/YFA (F-15 SPO)	1
HQ CSAF/RDQRM	1
HQ USAF/SAGF	1
Nav Air Test Cntr, SA-53A	1
USAF TFWC/TEM	1
AMSAA/DRXSY-J	1
AFATL/DLDG	1
DDC	2
AUL (AUL-LSE-70-239)	1
ASD/ENFEA	1
TAWC/TRADOCLO	1
AFATL/DLODL	9
AFATL/DL	1
HQ USAF/SAMI	1
Ogden ALC/MMWM	2
AFIS/INTA	1
ASD/ENESS	1
HQ TAC/DRA	1
HQ USAFE/DOQ	1
HQ PACAF/DOO	1
AFATL/DLODR	1
TAC/INA	1
ASD/XRP	1
US Army TRADOC Sys Analy Act/ ATAA-SL (Tech Lib)	1
COMIPAC/I-232	1
NWC/Code 40704	1

(The reverse of this page is blank)